Self-Learning Enhancement of the Measurement Quality for Electric Power Trains

Jakob Pfeiffer

Lehrstuhl für Datenverarbeitung
Technische Universität München

Self-Learning Enhancement of the Measurement Quality for Electric Power Trains

Jakob Pfeiffer

Vollständiger Abdruck der von der Fakultät für Elektrotechnik und Informationstechnik der Technischen Universität München zur Erlangung des akademischen Grades eines

Doktor-Ingenieurs (Dr.-Ing.)

genehmigten Dissertation.

Vorsitzender: Prof. Dr. rer. nat. Franz Kreupl

Prüfer der Dissertation:

1. Prof. Dr.-Ing. Klaus Diepold
2. Priv.-Doz. Dr.-Ing. habil. Dirk Wollherr

Die Dissertation wurde am 23.11.2020 bei der Technischen Universität München eingereicht und durch die Fakultät für Elektrotechnik und Informationstechnik am 18.04.2021 angenommen.

Jakob Pfeiffer. *Self-Learning Enhancement of the Measurement Quality for Electric Power Trains.* Dissertation, Technische Universität München, Munich, Germany, 2021.

Chair of Data Processing, Technische Universität München, 80290 München, Germany, https://www.ei.tum.de/ldv/.

„Ihr kennt die Welt nicht, Ritter. Was man *scheint*,
Hat jedermann zum Richter; was man *ist*, hat keinen."

Elisabeth in Schillers *Maria Stuart*

Acknowledgments / Danksagungen

Nach einem erfüllten Jahrzehnt an der Technischen Universität München endet mit dieser Arbeit meine akademische Laufbahn. In den vergangenen Jahren haben einige Menschen mein Leben bereichert, denen ich an dieser Stelle danken möchte.

Zuallererst gebührt mein Dank Prof. Dr.-Ing. Klaus Diepold dafür, dass er sich des Themas angenommen und mich immer mit seiner pragmatischen Art unterstützt hat. Seine Erfahrung half mir, die richtigen Meilensteine zu setzen, um die Arbeit zu einem erfolgreichen Ende zu bringen. Für seine hilfreichen Ratschläge bin ich sehr dankbar und die vielen Diskussionen weit über den Umfang dieser Arbeit hinaus haben mich nachhaltig inspiriert.

Genauso gilt mein Dank meinem Betreuer Christopher David, der sich stets in meine Probleme einfühlte, als wären es seine eigenen und mich enorm bei der Lösungsfindung unterstützte. Ohne seine Hilfe bei diversen Tools und Prozessen hätte ich die Arbeit sicherlich nicht in der relativ kurzen Zeit und in diesem Umfang geschafft.

Ferner möchte ich meinem Mentor Dr. Thomas Stauner danken. Die regelmäßigen Gespräche mit ihm halfen mir dabei, ein paar Schritte zurückzugehen und die Arbeit mit der nötigen Distanz als großes Ganzes zu betrachten. Nicht zuletzt sein Ratschlag, genau zu überlegen, welches Paper ich auf welcher thematisch passenden Konferenz einreichen soll, hat entscheidend zum Erfolg der Arbeit beigetragen.

Ohne meine Koautoren wäre die Arbeit nicht in der vorliegenden Form zustande gekommen. Deswegen gilt mein ganz besonderer Dank Peter Wolf, Roberto Pereira, Xuyi Wu, Ahmed Ayadi und Mohamed Ali Razouane.

Gerade in der Anfangsphase hat mich Marcel Ewers mit krEAtiven Methoden sehr bei der Themenfindung und im späteren Verlauf mit wertvollen Ratschlägen zur effektiven Präsentation der Ergebnisse unterstützt. Herzlichen Dank dafür. Ebenso gilt mein Dank Michael Böckl für die Initiierung der Doktorandenstelle. Bei allen Mitarbeiter(inne)n der Funktionsentwicklung und Applikation im E-Antrieb bei der BMW Group möchte ich mich bedanken für die stete Hilfe in technischen und prozessualen Fragen und für die gute Aufnahme im Team.

Ein ebenso großer Dank gilt den vielen Mitstreiter(inne)n im Doktorandenkreis der BMW Group. Der gegenseitige Erfahrungsaustausch sowie die Hilfe untereinander bei Problemen waren eine unverzichtbare Stütze bei der Bewältigung der ProMotion. Ich verzichte aufgrund der Fülle an mir wichtigen Menschen hier auf eine Nennung einzelner Namen; auch aus Sorge, einzelne zu vergessen. Ich denke, dass sich alle Betroffenen angesprochen fühlen. Die gemeinsamen Stammtische werde ich sehr vermissen.

Gleichmäßig vermissen werde ich meine zweite Heimat neben der BMW Group am Lehrstuhl für Datenverarbeitung. Die gerne auch kritischen Diskussionen am Mittagstisch haben mich sehr bereichert. Ich danke allen Doktorand(inn)en und Mitarbeiter(inne)n dort für die auch hier sehr gute Aufnahme ins Team und die positive Stimmung, mit der Montage für mich zum schönsten erdenklichen Start in die Woche wurden. Meinem zeitweiligen Büronachbarn Marius Hassler danke ich für die gute und konzentrierte Zusammenarbeit. Ganz besonders danke ich Ricarda Baumhoer, dass sie weder Aufwand, noch Mühen gescheut hat, um mir bei bürokratischen Hürden schnell und unkompliziert zu helfen.

Abschließend gilt mein Dank meinen Freunden und meiner Familie: Meinen Freunden und meiner Schwester Evelyn dafür, dass sie die letzten Jahre an und neben der Uni so lebenswert gemacht haben. Meinen Eltern für alles, was sie mir ermöglicht haben. Eduarda, für ihr stetes Antreiben, mich großen Herausforderungen zu stellen und nicht den leichten Weg zu gehen. Für ihre Hilfe, als ich zwischendurch den Glauben an ein positives Ende verloren hatte und für das Teilen der Freude über meine Erfolge.

Abstract

Amongst others, the cruising range of Electric Vehicles (EVs) suffers from overcautious High Voltage (HV) current restrictions during recuperation to prevent damages caused by deviations between measured and real values. During acceleration, the HV current restrictions limit the EV's performance unnecessarily. Decreasing the gap between measured and real values of HV currents thus serves as basis to increase the performance as well as the electric cruising range.

My goal is to increase the measurement quality of electric power trains. To keep the manual calibration effort as low as possible, the correction is executed with self-adjusting methods from the field of Machine Learning. The measurement correction shall be feasible for the standard hardware of common series EVs. This means that I correct measurement values based only on the information carried by measurement signals without additional sensors. Furthermore, all considered algorithms shall be feasible for execution on automotive Electronic Control Units (ECUs).

To correct the measurement deviations, I regard two sub-problems separately. To solve the problem of deviations caused by time delays, I compare several algorithms for Time Delay Estimation (TDE) and simulate not yet received signals with time series prediction. A fleet-based approach is my solution of choice to detect deviations caused by measurement inaccuracies with the help of measurement models and a classifier. The correction is then executed with Compressed Sensing.

The results of this work show that time delay correction can reduce measurement deviations from 25 % to below 5 % of the maximum current. The additionally developed fleet-based measurement correction is able to distinguish between hardware and measurement faults. Measurement faults caused by drifts and biases can be corrected with a recovery rate up to 90 %.

My results show that Machine Learning methods can successfully be applied to correct HV current measurements efficiently enough for automotive ECUs. However, especially when it comes to retrieve accurate corrections as efficiently as possible, manual calibration is required. It cannot be replaced, but rather be transformed by Machine Learning.

Kurzfassung

Die Reichweite von Elektrofahrzeugen leidet unter anderem unter übervorsichtigen Beschränkungen des Hochvoltstroms (HV-Stroms) zur Vermeidung von Schäden durch Abweichungen zwischen gemessenen und realen Werten während der Rekuperation. Während der Beschleunigung begrenzen die HV-Strombeschränkungen die Leistung des Elektrofahrzeugs unnötig. Das Verringern der Lücke zwischen gemessenen und realen Werten von HV-Strömen dient somit als Grundlage zur Steigerung sowohl der Leistung als auch der elektrischen Reichweite.

Mein Ziel ist es, die Messqualität von elektrischen Antriebssträngen zu erhöhen. Um den manuellen Applikationsaufwand so gering wie möglich zu halten, erfolgt die Messkorrektur mit selbstparametrisierenden Methoden aus dem Bereich des Maschinellen Lernens. Die Messkorrektur soll auf der Standardhardware gängiger Serienelektrofahrzeuge ausführbar sein. Das bedeutet, dass ich die Messwerte nur auf Grundlage der von den Signalen getragenen Messinformationen ohne zusätzliche Sensoren korrigiere. Darüber hinaus sind alle in dieser Arbeit betrachteten Algorithmen für die Ausführung auf in der Automobilindustrie üblichen Steuergeräten ausführbar.

Um die Messabweichungen zu korrigieren, betrachte ich zwei Teilprobleme getrennt. Das Problem der durch Zeitverzögerungen verursachten Abweichungen löse ich, indem ich mehrere Algorithmen zur Zeitverzugserkennung vergleiche und noch nicht empfangene Signale mit Zeitreihenvorhersage simuliere. Durch Messungenauigkeiten verursachte Abweichungen identifiziere ich mit Hilfe eines flottenbasierten Ansatzes unter Verwendung von Messmodellen und eines Klassifikators. Die Korrektur führe ich dann mit Spärlichkeitsannahmen und Komprimierter Erfassung aus.

Die Ergebnisse der vorliegenden Arbeit zeigen, dass die Zeitverzögerungskorrektur die Messabweichungen von 25 % auf unter 5 % des maximalen HV-Stroms korrigieren kann. Die zusätzlich entwickelte flottenbasierte Messkorrektur ist in der Lage, zwischen Hardware- und Messfehlern zu unterscheiden. Messfehler wie Relativ- und Absolutfehler können in der vorliegenden Arbeit mit einer Wiederherstellungsrate von bis zu 90 % korrigiert werden.

Meine Ergebnisse zeigen, dass Methoden des Maschinellen Lernens erfolgreich angewendet werden können, um HV-Strommessungen mit der für Automobilsteuergeräte notwendigen Effizienz auszuführen. Dennoch wird die manuelle Applikation weiterhin benötigt. Vor allem dann, wenn es darum geht, präzise Korrekturen effizientestmöglich auszuführen. Manuelle Applikation kann durch Maschinelles Lernen nicht ersetzt, sondern lediglich transformiert werden.

Contents

1 Introduction

One of the biggest challenges humanity is facing in these times is the Global Warming. To slow it down, the emission of greenhouse gases must be reduced [21]. The reduction can be reached with the help of decarbonization. One of the most promising approaches for decarbonizing individual transport are Electric Vehicles (EVs) [21].

However, sales numbers of EVs are still small compared to Internal Combustion Engine Vehicles (ICEVs) [18]. One of the reasons for the reluctant buying behavior is the so-called *range anxiety*. Range anxiety describes the behavior of people not purchasing an EV because of being afraid to break down during a trip due to a completely discharged High Voltage Battery (HVB) [18].

There are two ways to prevent such a break down. For example, one can increase the capacity of the HVB. Although relatively simple from a technological point of view, this approach has several drawbacks. Besides not being sustainable, increasing the HVB would mean to increase the costs for the production of the anyway most expensive component (up to 45 % of the total price) of an EV [1]. The resulting increased purchasing price would further hinder fast EV adoption by the markets. A more sustainable and cost-effective alternative is to increase the efficiency of the electric power train.

1.1 Motivation

The efficiency of the electric power train suffers from measurement deviations. The measurements recorded in the scope of this thesis show that deviations between measured and real values can be up to 25 % of the maximum battery current (compare Figure 1.1). The deviations lead to several restrictions. For example, the battery current is restricted, when the power train is operating close to its physical limits. Harming the limits would lead to severe damages in the HVB [11, 13]. The damages would shorten its lifetime or, in the worst case, lead to a destruction of the HVB. To prevent such damages, the electric power train restricts the maximum current as shown in Figure 1.2a with the help of so-called *battery protection limits*. The limits may not be harmed even under the worst measurement conditions. To guarantee safe power train operation in every situation, the maximum possible measurement inaccuracy must be considered and added as additional offset to the battery protection limits (see Figure 1.2b).

However, these offsets also have disadvantages. For example, when the EV is accelerating, too conservative offsets restrict the vehicle's performance unnecessarily.

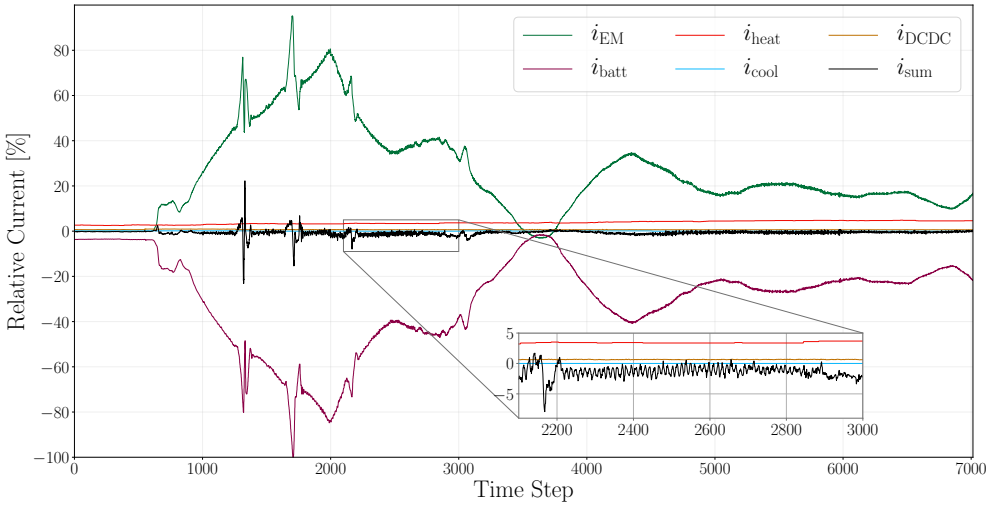


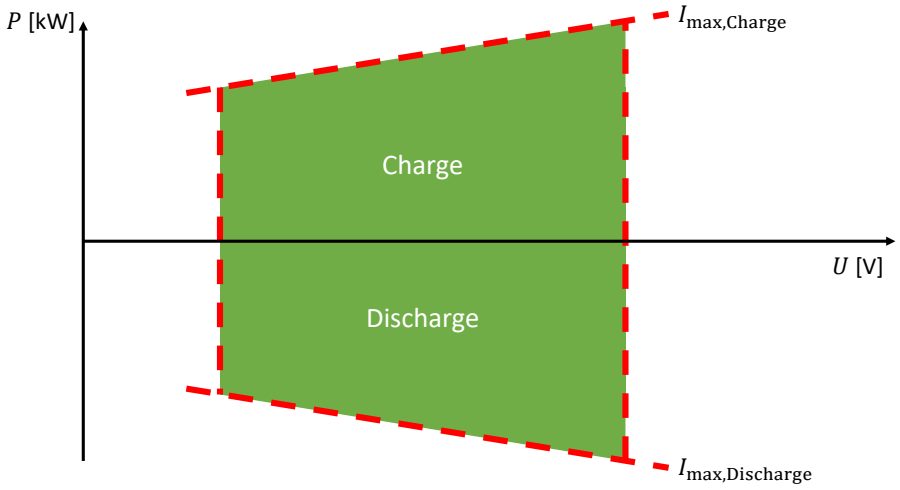
Figure 1.1: Measurements of all High Voltage (HV) currents of an EV during a test drive [19]. The sum of all currents must be equal to 0 % according to Kirchhoff's current law. The measurements show that the deviation between the sum of the measured currents (black) and its real value (0 %) is up to 25 % of the maximum current. Zooming into the measurements reveals that the measurement deviation is higher than the overall consumption of whole HV components like the DC-DC converter. The noise spectrum of the deviation is half as high as the second highest current consumption of this drive i_{heat} [19].

During the opposite case, when the EV is decelerating, too conservative offsets restrict the energy recuperated to the HVB. As a result, the power train's efficiency and thus the vehicle's electric cruising range decrease. The decrease can be avoided by improving the measurement quality. Better measurements would allow to minimize the battery protection offsets and thus increase the performance as well as the efficiency and the cruising range of EVs.

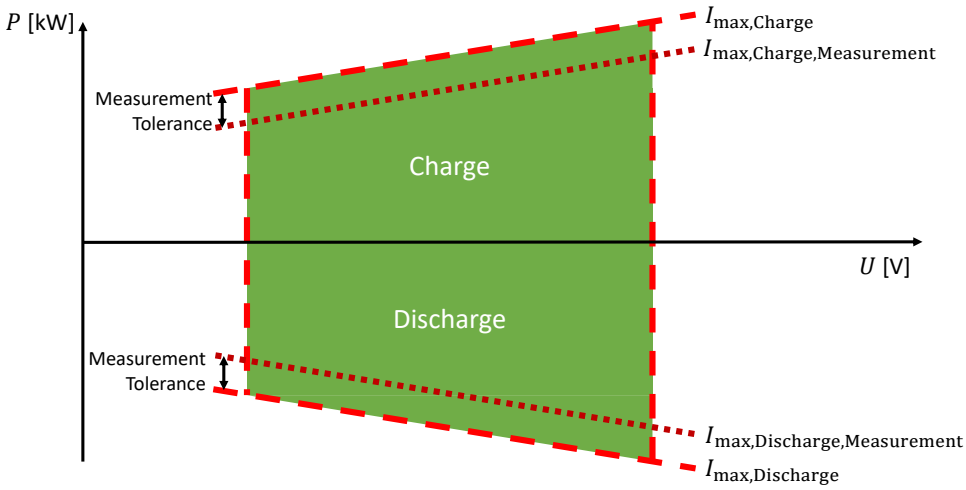
1.2 State of the Art

This section introduces the state of the art concerning the development of EVs. The development can be distinguished between hardware and software development. The hardware of an EV is described in subsection 1.2.1, where I present all HV components of the electric power train. In subsection 1.2.2, I present the hardware layout of vehicular processing units. The function and software development process in the automotive industry is introduced in subsection 1.2.3.

In the scope of this work, the term EV denotes vehicles which are able to drive purely electric. Examples for these vehicles are Battery Electric Vehicles (BEVs) like the Mini Cooper SE or Plug-in Hybrid Electric Vehicles (PHEVs) like the BMW 740Le.



(a) General battery protection limits according to Kowallik [13].



(b) Battery protection limits with additional offsets considering measurement deviations.

Figure 1.2: Battery protection limits for charging and discharging a HVB.

1.2.1 HV Components of an Electric Power Train

Usually, electric power trains consist of the five HV components presented below. Thereby, each component can have multiple units installed in one vehicle. In the Audi e-tron 55 quattro, for example, two Electric Machines (EMs) are installed. An overview of all HV components is pictured in Figure 1.3 which shows the electric power train of a BMW i3 as an example.

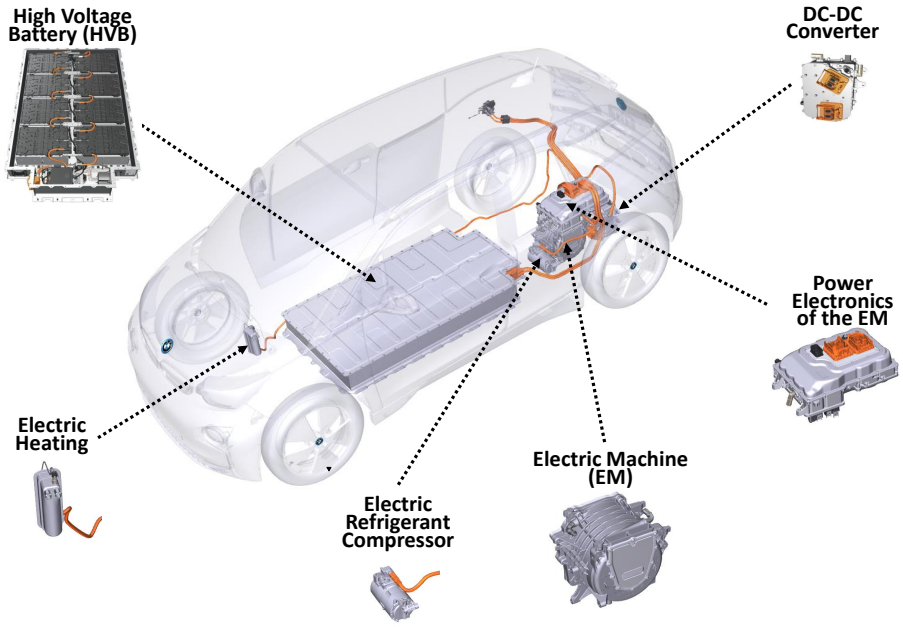


Figure 1.3: Power train of a BMW i3 with all HV components considered in this thesis.

The Electric Machine (EM)

The EM is the heart of the electric power train. It can be used both as a motor for accelerating and as a generator for recuperating kinetic energy during braking [11]. Therefore, it can be seen both as a sink as well as a source of electric power.

The EM is coupled with the power electronics. In motor mode, the power electronics convert direct current into phase currents for the operation of the EM. In generator mode, this conversion is inverted. From the perspective of power sinks and sources, EM and power electronics form a unit. Therefore, when speaking of the EM, I always mean both the EM and the power electronics in the scope of this thesis. The same is valid for other HV components which make use of power electronics.

The High Voltage Battery (HVB)

The HVB is the central energy storage system. It can charge or discharge electric power at any time step. This means that it too can be used as a source and sink of power. When charging or discharging the HVB, it is always important to respect the battery protection limits (see Figure 1.2) in order to prevent the HVB of overheating, over and deep discharge. The prevention is crucial for guaranteeing the longest possible service life and highest safety of the HVB [11].

The Electric Heating

The electric heating is used to warm up the passenger compartment and, if necessary, vehicular components. Like the other HV components, the electric heating is connected to the HV circuit [11]. It is a pure sink of electric power.

The Electric Refrigerant Compressor

The electric refrigerant compressor is the counterpart to the electric heating. It is also supplied with power via the HV circuit [11]. The refrigerant compressor is used for cooling power train components and the passenger compartment. Like the electric heating, it is a pure power sink.

The DC-DC Converter

The DC-DC converter supplies the conventional low-voltage onboard grid with electrical power. To this end, it transforms the adjacent HV from usually $90\,\mathrm{V}$ to $800\,\mathrm{V}$ to a Low Voltage (LV) of $12\,\mathrm{V}$ for passenger cars, $24\,\mathrm{V}$ for trucks or $48\,\mathrm{V}$ for some modern vehicles [11]. The EVs considered in the context of this thesis have a rated voltage of around $400\,\mathrm{V}$ in the HV range and $12\,\mathrm{V}$ in the LV range. All here considered EVs use the DC-DC converter only as power sink.

1.2.2 Vehicular Information Processing Topology

Most of the automobiles purchased today do not have a central computer. Instead, they have distributed systems consisting of several dozens Electronic Control Units (ECUs) [25]. The ECUs are connected to each other and communicate via bus networks.

Electronic Control Units (ECUs)

Originally introduced to control the combustion process and thus reduce fuel consumption and pollutant emissions, there are now many areas of application for automotive ECUs [12]. For example, safety-critical systems such as Anti-lock Braking Sys-

tem (ABS) and Dynamic Stability Control (DSC) are integrated into the vehicle with the help of ECUs [12, 22].

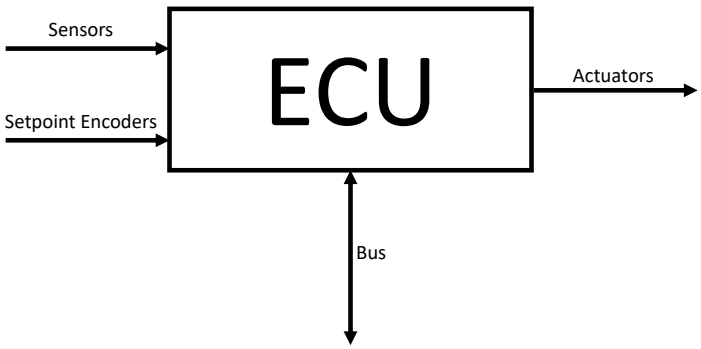


Figure 1.4: Interfaces of an ECU according to [22], greatly simplified from [7].

As depicted in Figure 1.4, ECUs receive input from sensors and setpoint encoders [22, 7]. Based on these inputs, the ECUs execute software functions during operation of the vehicle [22]. Today, the number of functions performed by a control unit is typically in the three-digit range [22]. Depending on the functions, the ECUs control actuators accordingly [22, 7].

With the number of features and the sensor signals to be processed in modern vehicles, the demands on the performance of the ECUs also increase. Special high performance ECUs are developed for the field of application of autonomous driving [25]. However, these high performance ECUs are very often in a prototypical development stage [25]. Due to their advanced processors, these ECUs are more expensive than common ECUs.

Most of the ECUs available today, especially those applied to the power train of EVs, do not offer such a high computation performance. Therefore, high performance ECUs are not considered in this thesis and my requirement for the proposed solutions is that they can be executed efficiently.

Bus Communication

While the potential of ECUs was limited in the beginning to the connected sensors, since the 1990s there has been a huge increase in the possibilities of use by networking the control units with each other with the help of buses [22]. Buses are time-synchronized and allow scalable exchange of information while reducing the required cables [22, 12]. One of the most famous bus systems is the Controller Area Network (CAN) bus [12, 22]. It is the basis of many modern bus systems [12]. Nevertheless, there are many other bus systems, some of which are specialized in specific

applications [22]. In addition, it is common for certain information from ECUs to be placed on multiple buses to enable advanced features that require a lot of different information [12]. An exemplary bus network is shown in Figure 1.5.

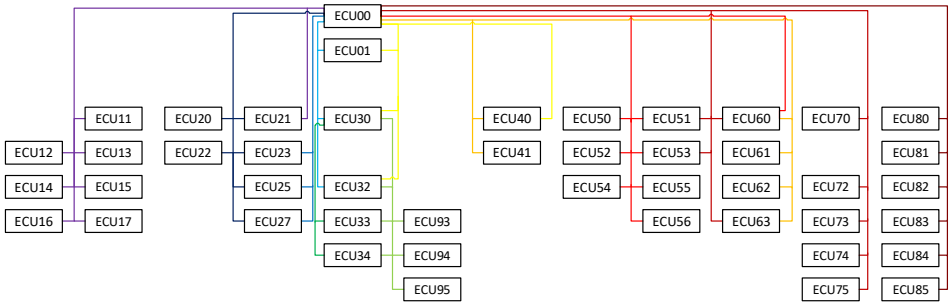


Figure 1.5: ECUs (labeled boxes) and buses (colored lines) of a PHEV BMW 745Le xDrive. Since the exact number of ECUs and buses depends on the individual optional equipment, this figure can only serve as an example. The exact number of ECUs and maybe buses may vary from vehicle to vehicle, including the same model.

1.2.3 The Function and Software Development Process in the Automotive Industry

In the automotive industry, software development is carried out according to the so-called *V-model*. The V-model is diagrammed in Figure 1.6. The model elements *functional development* and *application* are the focus of the present work.

Variant Management with Modular Systems

Automotive development is characterized by a strong variety of models and variants today. In the past, there were comparatively few models of a manufacturer with slightly different characteristics. In 1960, for example, only three different car models were available from the BMW Group. Today, 60 years later, there are around 50 models. In addition, there are different variants and equipment. In order to minimize the development effort for such a number of models, *modular systems* are used. These allow the use of individual components, the so-called *modules*, in several models with one-time development effort [24, 22]. In the case of the function modules considered here, this means that once developed (partial) functions of the electric power train are used in several vehicle models and variants. The appropriate parameterization of the

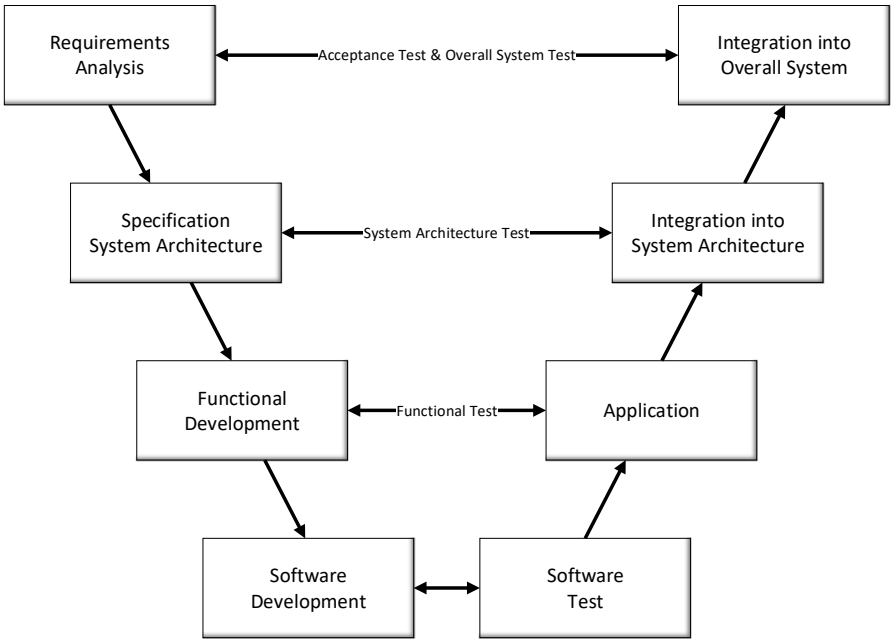


Figure 1.6: Simplified V-model according to [22] and [21], as it is used in the automotive industry.

respective functions takes place in the application (see Figure 1.7). By using the function blocks, the additional effort for the functional development is minimized with an increased number of models and variants.

Functional Development

Functional development in the sense of this work describes the development of function modules. This is done in several steps. First, the requirements arising from the specification of the system architecture are reviewed and further developed [22]. Functional structures are then developed together with the system architects. In these structures, total functions are divided into subfunctions according to logical and physical aspects [24].

The main tasks of functional development include the search for principles of action and solution variants and selecting and evaluating suitable algorithms to solve the specified task [24]. Above all, such algorithms are preferable, which allow different variants only by different parameterizations without changing the functions [24].

Together with the applicators and function testers, function developers create test cases and test the correct implementation and expression of the functions [22].

The final task of functional development is to create designs and requirements that

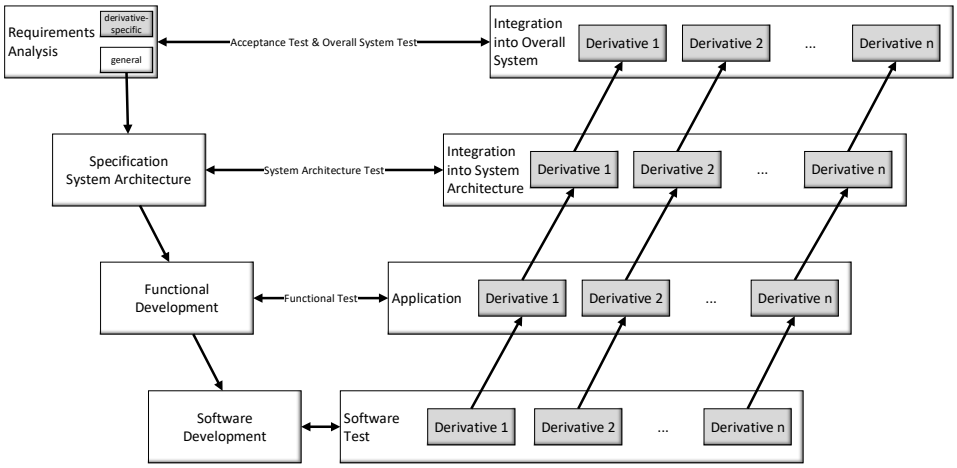


Figure 1.7: The V-model from Figure 1.6 taking into account modular-based development with different variants.

serve as the basis for software development [24, 22]. Software development is the next step in the development process (see Figure 1.6), in which the functions are implemented.

Application

The application or calibration refers to the finding and setting of the optimal parameters of the developed functions. The parameters can be characteristic values, characteristic curves or characteristic fields. Often, as shown in Figure 1.7, the parameters must be applied individually for each vehicle model and all different variants of a model [22]. This work is carried out by so-called *applicators* or *calibration engineers* either directly on the vehicle, at test benches or in simulation environments [5]. The number of parameters to be set usually increases with each model generation due to new functions. For modern electric vehicles such as those studied in the scope of this work, more than 40,000 parameters are currently standard.

1.3 Scope of the Thesis

The goal of this thesis is to minimize the deviation between measurements and real values of HV currents in electric power trains. To minimize the effort for the applicators, my goal is to achieve a solution with self-learning methods. To enable solutions which are feasibly applicable to the software development processes in the automotive industry, I work with real measurement data from close to production vehicles without additional sensors.

Thereby, this thesis answers the following research questions.

RQ1 Can the deviation between measured and real values of the HV currents in the power train of EVs be corrected?

RQ2 Can this correction take place without additional sensor systems based only on the measurements available in modern series EVs?

RQ3 What Machine Learning methods can be applied to automotive ECUs to solve the problems described above?

RQ4 Can the automated correction be implemented without or at least with minimal additional effort for the applicators?

To answer the research questions, I work with the following hypotheses.

H1 The deviation between measured and real values of the HV currents can be corrected, if the problems of measurement inaccuracies and time delays are considered separately.

Hypothesis H1 is reflected by the thesis as follows. While sections 4.1 and 4.2 deal with the detection and correction of measurement inaccuracies, the sections 3.1 and 3.2 focus on the correction of measurement deviations caused by time delays.

H2a For measurement deviations caused by time delays, the correction can work without additional sensors. With the help of *Kirchhoff's current law*, the deviation can be minimized by shifting slower signals to the past.

H2b For measurement deviations caused by measurement inaccuracies, *Kirchhoff's current law* alone is not sufficient. To identify the *ground truth* underlying the measurements, we additionally need a redundant measurement to verify the plausibility of the HV current measurements. This redundant measurement can be provided by a fleet of many equally constructed EVs. For a sufficient large number of EV, the averaged HV current measurement will tend to the real value.

I focus on hypothesis H2a in section 3.1. In the sections 4.1 and 4.2, I show the feasibility of H2b. The hypothesis is justified due to the following reasoning.

Clearly, if the whole fleet for a large number of EVs measured on average e.g. an offset fault of -5 A, this would be a systematic measurement fault. In the contrary case, if it was not a systematic fault, the mean measurement fault of the fleet would converge towards 0. In my thesis, I concentrate on non-systematic measurement faults and show how to correct them. Non-systematic in this context describes the whole fleet. Nevertheless, from the perspective of an individual vehicle, the error might be systematic. The problem of systematic measurement faults of the whole fleet is not the scope of my thesis. That problem can be solved otherwise with the help of quality

instruments. For example, at the end of the production line, the vehicles as well as the individual components and sensors have to successfully complete several tests. There, systematic measurement faults would be recognized and the component supplier would be further investigated to solve the quality issue. My focus is on vehicles after they left the production line. Some sensors start to drift (which means slowly increasing measurement faults) with proceeding age. As the production life cycle of automobiles commonly is around 7 years and the majority of sensors usually works faultless for very long time, the fleet of vehicles of the same model will always be balanced. Even if the mean of the measurement faults would not sum up to 0, it would be negligibly small compared to the majority of healthy (and in most cases younger) vehicles.

H3 Machine Learning methods allow appropriate solutions for the problems described above. However, for the sake of feasible implementation on automotive ECUs, the algorithms must be chosen carefully with regard to the required memory consumption and runtime.

This hypothesis is confirmed in all papers in chapters 3 and 4. In each paper, I compare the runtime of all considered algorithms and try to achieve the best compromise between runtime and accuracy. All solutions proposed in the papers are selected to minimize the amount of data that needs to be stored on-board of the EV.

H4 With the help of self-adjusting methods from the field of Machine Learning, the automated measurement correction can be implemented with minimal additional effort for the applicators.

Hypothesis H4 is also confirmed in all papers printed in chapters 3 and 4. To keep the additional effort for the applicators as low as possible, I work only with self-adjusting methods from the field of Machine Learning. For each sub-problem, I compare several algorithms to select the most suitable one, depending on the requirements.

1.4 Structure of the Thesis

After introducing the research topic in chapter 1 above, I show the current state of the research in chapter 2. There, I cite literature addressing related research topics, accentuate the differences to my work and formulate the contribution of this thesis. The solutions implemented in the scope of this thesis and the corresponding results are shown in the previously published papers printed in chapters 3 (measurement deviations caused by time delays) and 4 (deviations caused by measurement inaccuracies). In chapter 5, I summarize the most relevant points of my work.

2 Related Work and My Contribution

My contribution in this thesis is the development of an automated measurement correction. The proposed correction is able to detect and correct deviations between HV current measurements and real values in the power train of EVs. The deviations can be caused by measurement inaccuracies as well as time delays in the communication between the HV components. The correction takes place without additional hardware and bases only on the existing measurement signals alone. Due to its low computational effort, it is feasible to be executed on the existing ECUs of modern production vehicles. To the best of my knowledge, this is the first time – besides the already published papers in the scope of this thesis – that such a measurement correction system is proposed for the standard hardware of power trains of EVs.

Therefore, I am not able to compare other researchers' experimental EVs to the ones I used to collect and improve measurements in the scope of this thesis. Instead, I compare different experimental vehicles from the application field of assisted and autonomous driving. Similar to the electric power train, these applications require accurate measurements of high quality to control the vehicles safely in real time. One of the outstanding events of research into autonomous driving is the *DARPA Urban Challenge 2007*. There, the spearheads of the international research community compete against each other in autonomously driving through a parkour with several experimental vehicles. Figures 2.1 and 2.2 show the vehicles participating in the challenge for the Massachusetts [14, 15] and the Karlsruhe Institute of Technology [10] respectively.

The experimental vehicle (see Figure 2.3) of Jo et al. does not participate in the DARPA Urban Challenge. Nevertheless, the authors' goal is to enable autonomous driving, too. For a sufficiently precise localization, they present an approach for correcting bias faults in Global Positioning System (GPS) measurement data [9]. Like the two vehicles mentioned above, there are several additional sensors mounted to the vehicle of Jo et al.

Looking at the vehicles, it is conspicuous that it is very common among the researchers to achieve a higher measurement quality by integrating the maximum of available sensors and processing units into the vehicles. Although the approach of increasing mounted hardware is quite expensive from a cost-oriented perspective, it is trivial from the problem view. Of course, the measurement quality increases by adding more and better sensors and providing additional resources to process the measurement data.

This attitude contradicts my approach. My goal is to achieve the highest possible measurement quality with the existing measurements of series sensors in close to pro-

Figure 2.1: Talos, the experimental vehicle which participated for the Massachusetts Institute of Technology in the DARPA Urban Challenge [14]. Its sensor equipment is far beyond the series sensor set of the basic vehicle Land Rover LR3 [15].

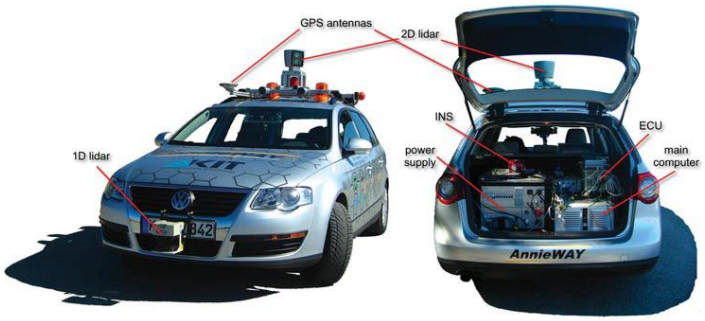


Figure 2.2: AnnieWAY, the vehicle that participated in the DARPA Urban challenge for the Karlsruhe Institute of Technology [10]. The image shows on the left side the additional sensors mounted on the experimental vehicle. On the right side, you can see the additional computation units installed in the vehicle, because common ECUs would not be capable to execute the algorithms proposed by the researchers in real-time.

Figure 2.3: The experimental vehicle of Jo et al [9]. To enable precise GPS localization, the researchers mount several additional sensors on the vehicle. Amongst the GPS sensors marked with red circles, the figure also reveals additional Radio Detecting And Ranging (RADAR) and Light Detecting And Ranging (LIDAR) sensors installed at the front and rear bumper.

Figure 2.4: One of the EVs used to record measurement data for the experiments of this thesis. Besides data loggers for recording measurement and bus data, no additional signal processing hardware beyond the scope of a series vehicle is mounted[1].

duction vehicles. When processing information of the measurements, I pay attention to ensure that the algorithms I use can be realistically carried out on serial ECUs. In contrast to the experimental vehicles mentioned above, the ones used in the scope of this thesis are equipped only with sensor sets close to series production as shown in Figure 2.4[1]. Here, the only additionally mounted hardware is used to log measurement data to enable simulation on computers outside the vehicle after driving.

Another field of application of measurement correction are the power trains of ICEVs. In contrast to EVs, Machine Learning approaches are quite common to increase the measurement quality there.

For example, Froschhammer applies Neural Networks to internal combustion engines in his very inspiring dissertation [6]. The author uses Recurrent Neural Networks to automatically detect and correct measurement deviations in the injection system of the power train. In the present thesis, my goal is to extend the idea of automated measurement deviation detection and correction to electric power trains.

In a related dissertation, Beuschel introduces another example of Neural Networks [2]. With the help of Harmonically Activated Neural Networks, the author compensates time delays of measurement signals and hardware deviations. His focus is on an optimal cylinder running behavior, which is not necessary for EVs. Nevertheless, the idea to use Machine Learning methods to identify hardware faults and compensate time delays inspires my work and is extended here to electric power trains.

Lu et al. propose an inspiring approach for measurement deviation detection and correction for aero engines [17]. Their correction is able to identify different kinds of measurement faults with the help of an Online Sequential Extreme Learning Machine with Memory principle (MOS-ELM). Unfortunately, the training time of several seconds for a MOS-ELM is too long to apply it directly to an EV for online learning during execution. Nevertheless, Lu's article, especially the distinction between drift and bias faults, is very helpful for my work.

Although exemplarily applied to power trains of ICEVs, Wolf et al. introduce a quite general approach for automotive diagnostics [26]. It is not restricted to individual components but rather analyze data independent from its origin. With the help of Long Short-Term Memory Neural Networks (LSTMs), the authors analyze vehicle data. Deviations in the data lead them to unusual behavior and allow conclusions about possible faults. In the paper printed in section 4.1, Wolf and I show that their approach can be extended to electric power trains. Together with Pereira, we additionally implement and evaluate State-Space Models as alternative to LSTMs.

If Machine Learning methods are used to optimize EVs, the focus is usually on energy management strategies.

For example, Li et al. present an interesting approach of Deep Reinforcement Learn-

[1] I downloaded the image from `https://www.press.bmwgroup.com/global/article/detail/T0293467EN/electromobility-under-extreme-conditions:-the-bmw-ix3-the-bmw-i4-and-the-bmw-inext-undergo-cold-testing-in-the-arctic-circle?language=en` on the 25th of March 2019.

ing for the power trains of PHEVs [16]. Their goal is not to correct measurements but to develop an energy management strategy for the optimal operation of the internal combustion engine. Similar to my work, Li et al. pay attention to the runtime of their algorithms to enable practical applications of their approach. Although the authors specify the runtime of their approach, they unfortunately do not mention the computation hardware used.

Cussigh and Hamacher have a similar goal like Li et al. Their goal is to find optimal driving and charging strategies for BEVs on long distance trips [3]. Unlike Li et al., who try to optimize the energy consumption during a trip, Cussigh and Hamacher focus on the minimization of the time required for traveling and charging.

The approach proposed by Straub et al. is very similar to the one proposed in this thesis [23]. The authors' goal is to predict the energy consumption of BEVs accurately with Machine Learning. The Machine Learning methods implemented by the authors are regression and classification, similar to my approaches in sections 4.1 and 4.2. Another similarity is that Straub et al. also work with fleet data. However, different to my work, the authors' interest lies on the energy consumption of the whole vehicle. They do not consider explicitly individual HV components.

There are also works on improving measurements of electric power trains. However, these works only focus on individual HV components. Xiong et al. give an overview of State of Charge (SoC) estimation methods for HVBs [27]. HVBs are the only application area. The methods considered can only be transferred to other HV components in exceptional cases, if at all. In contrast, the methods presented in my work can be flexibly applied to all HV components.

Independent from automotive applications, Ren et al. develop a calibration method for current sensors in the electrolysis industry [20]. Different to our circumstances, Ren et al. deal with relatively high currents. The current values in the automotive domain for are usually much smaller. Another difference is that, in contrast to Ren's system, automotive HV systems usually do not consist of redundant sensors.

Hamrita et al. propose a generic measurement correction approach for voltage instrument transformers [8]. According to the authors, the approach can also be extended to current instrument transformers. With the help of a model of the transformer, their approach is able to correct measurements of steady-state waveform signals with repetitive learning control. Unfortunately, the HV currents of electric power trains are not necessarily wave-shaped, which makes this approach unfeasible for our problem.

3 Time Delay Correction

In order to reduce the deviations between real and measured HV currents in the electric power train and to answer the questions formulated in section 1.3, I split the main problem into two sub-problems. While the focus of chapter 4 lies on deviations caused by inaccuracies in the measurements, the following sections concentrate on measurement deviations caused by time delays. In section 3.1, I show how to detect time delays between the measurement signals. With the knowledge of the emerging time delays, the resulting measurement deviations can be corrected directly. However, the information of not yet received measurements cannot be reconstructed only by knowing the value of the time delay. Section 3.2 presents a suggestion on how to draw conclusions about the missing information.

3.1 Time Delay Estimation

The following section consists of two papers. The first paper validates Hypothesis H1 by focusing on measurement deviations caused by time delays.

Consequently, Hypothesis H2a is validated by the solutions proposed in the paper. All solutions try to find the optimal time delay between two corresponding HV current measurements, eg. between HVB as source and another considered HV component as sink. The optimal time delay is determined as the one which minimizes the difference between the two measurement signals according to Kirchhoff's Current Law.

Hypothesis H3 is validated by the three evaluated methods Linear Regression, Variance Minimization and Adaptive Filters.

Regarding Hypothesis H4, it is positive to mention that all proposed methods are able to learn on the data alone without additional manual parametrization. However, since this is, in contrast to the approach proposed in 4.1 and 4.2, an online solution, there results some parametrization from the treatment of the data. For example, it might be useful to manually apply the length of the input data or the distance between input samples. Since the number of parameters to be applied manually depends strongly on the respective problem, the used algorithm and the HV component dependent properties of the measurement curve, unfortunately no general statement can be made here about the workload for the applicators.

My contribution to this paper is the fundamental problem statement, the recording and preprocessing of measurement data, the conceptualization of possible solutions as well as the choice of the considered methodologies. I implement the very first

versions of the Linear Regression and a Root Mean Square Error (RMSE) Minimization approach. Xuyi Wu's contribution is the further development of the RMSE Minimization approach to a Variance Minimization approach. Furthermore, the full implementation of the Adaptive Filters approach is her achievement. In the paper, she is responsible for the sections considering the experimental setups and the results. My responsibility are the sections regarding the introduction, the state of the art, the concepts, the discussion and the conclusion and outlook. Furthermore, I review the sections written by Xuyi Wu before submission and reformulate the sections to meet the reviewers' requirements after submission.

The 10th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications
18-21 September, 2019, Metz, France

Automated Time Delay Estimation for Distributed Sensor Systems of Electric Vehicles

Jakob Pfeiffer, Xuyi Wu
BMW Group, Petuelring 130, 80788 Munich, Germany, {Jakob.J.Pfeiffer, Xuyi.Wu}@bmwgroup.com
TU Munich, Chair for Data Processing, Arcisstr. 21, 80333 Munich, Germany, {Jakob.Pfeiffer, Xuyi.Wu}@tum.de

***Abstract* — Deviations between electric current measurements and reality can cause severe problems in the power train of electric vehicles (EVs). Among others, these are unnecessary power limitations and inaccurate performance coordination during driving or charging. One reason for these deviations are time delays. In this paper, we present three different approaches for time delay estimation (TDE) and evaluate these with real data from power trains of EVs. Besides the accuracy of the TDE, the focus of our evaluation lies on the computational efficiency to enable an execution on automotive electronic control units (ECUs). The Linear Regression approach suffers even from small noise and offsets in the measurement data and is unsuited for our purpose. A better alternative is the Variance Minimization approach. It is not only more noise-resistant but also very efficient after the first execution. Another interesting approach are Adaptive Filters presented by Emadzadeh et al. Unfortunately, Adaptive Filters do not reach the accuracy and efficiency of Variance Minimization in our experiments. Thus, we recommend Variance Minimization for TDE of current signals in the power train of EVs.**

Keywords* — *automotive; current; electric power train; electric vehicle; embedded systems; delay; detection; distributed systems; measurements; power train; sensor; signals; time delay estimation

I. Introduction

Kirchhoff's current law states that the sum of all currents in an electric system is equal to 0 A. However, considering measurement signals of electric vehicles (EVs) with distributed sensor systems, the sum of all currents can differ up to 20 % of the maximum current (see Figure 1). If we look closer at the root mean square error (RMSE) of the sum of currents $\mathrm{RMSE}(i_{\mathrm{sum}}) = 0,67\%$, we realize that it has the same value as the current of the DCDC converter which is $\mu_{i_{\mathrm{DCDC}}} = 0.67\%$ on average.

Another value than 0 A for the sum of all currents indicates that there is a divergence between measurements and reality. The divergence becomes problematic when the power train is operating close to the system boundaries. For example, there are boundaries for the protection of the high voltage battery (HVB). The HVB is only capable to discharge or charge a restricted amount of power. Higher amounts would threaten the HVB's lifetime and safety [1]. Additional offsets might be added to the boundaries to ensure a safe operation mode even for high divergences between measurements and reality, although

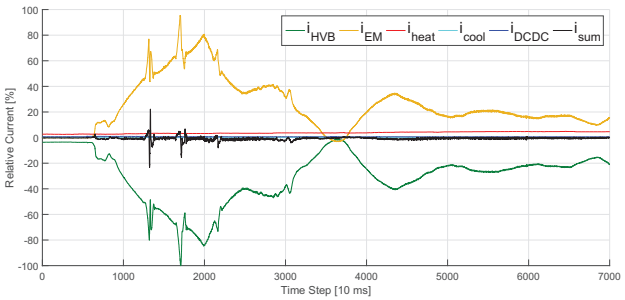


Figure 1. Currents of all HV components in an EV on a test drive. The sum of all currents i_{sum} is plotted in black. According to *Kirchhoff's current law*, it should be constantly 0 %. But looking on the measurements shows that the deviation i_{sum} is higher than the current of the DCDC converter i_{DCDC}. Even its noise spectrum is approximately half as high as the consumption of the heating i_{heat}, which is the second largest consumer in this drive.

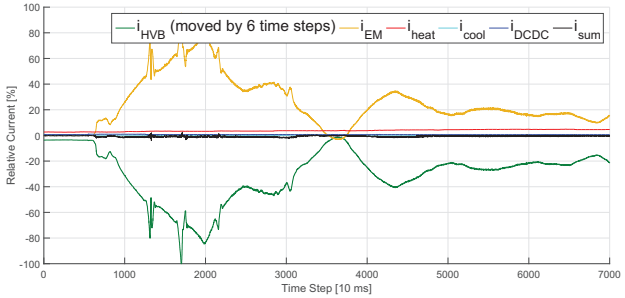


Figure 2. The same test drive like in Figure 1 but with the battery current i_{HVB} (green) shifted by 6 time steps. The sum of all currents i_{sum} (black) is significantly closer to 0 %.

they have some drawbacks. In the charging case, most notably during recuperation, the HVB might not allow the full power level, even though it would be capable to handle it. In the opposite case, the system might not release requested power, although the HVB could provide it in reality. As a result, the mentioned offsets restrict the cruising range and performance of the EV.

Besides measurement faults and sensor uncertainties, the divergence between measurements and reality is caused by time delays. The delays result from distributed sensor systems in the power train. The high voltage (HV) components have their own electronic control unit (ECU)

978-1-7281-4069-8/19/$31.00 ©2019 IEEE

which is connected with the current sensors and processes the sensor information. The ECUs exchange these information via bus systems. The buses require individual amounts of time to send the measurement signals. Thus, from an ECU's point of view, the sensor information from other ECUs arrives with individual delays. Figure 2 shows an example of the sum of all currents i_{sum} being reduced by shifting a signal by 6 time steps.

The aim of this work is to automatically detect the time delay between measurement signals from different sensors. For this purpose, we develop two different approaches. One of them is based on Linear Regression, whereas the other one optimizes the estimated variance of the difference between several signals. We compare our approaches to other state of the art time delay estimation (TDE) algorithms and evaluate them with a focus on precision and run-time efficiency. Apart from allowing a more accurate power distribution, the automated TDE helps to reduce the battery protection offset and thus to increase the performance and range of EVs.

The rest of this paper is structured as follows. Section II states related work and our contributions. In section III, we explain the theory behind our work before we describe the practical experiments in section IV. The results of our experiments are stated in section V. In section VI, we discuss the advantages and drawbacks of the proposed concepts. Finally, we draw our conclusions and give a short outlook in section VII.

II. State of the Art

There exists plenty of literature about TDE, although – to our best knowledge – none of them is tailored to the specific problem of TDE of current signals in EVs.

Zeng et al. [2] introduce a statistical approach to predict the delay of a bus message. The message's content does not need to be known to achieve high accuracy. This is different from our scenario where we want to make use of the information carried by the message. In contrast to Zeng et al., we do not require to predict the time delay accurately to milliseconds. For our purposes, an estimation of the number of delayed discrete time steps is sufficient.

Svilainis et al. [3] present another interesting approach. Their goal is to estimate the time passed between emitting an ultrasonic signal and absorbing its reflection. Like Zeng et al., they require high precision. Another difference to our approach is that their algorithms make use of the pulse form of ultrasonic signals. Our signal as plotted in Figure 1 can vary in a large range and does not necessarily contain pulses (e.g. after time step 5,000).

Emadzadeh et al. [4] show an inspiring approach for detecting the relative position of spacecrafts. For retrieving the position, they examine an X-ray signal received by two spacecrafts and determine the time delay between them. For the TDE, they use Adaptive Filters. This approach seems very promising to us. We implement the algorithms of Emadzadeh et al. and compare them to ours in order to find out, if their approach can be transferred from X-ray signals to current measurements in the power train of EVs.

Our contribution is the development of a regression-based and a Variance Minimization-based algorithm for TDE. We transfer the ideas introduced by Emadzadeh et al. to the domain of currents in the HV system of EVs and compare the results to our approaches in matters of accuracy and computational performance.

III. Concepts

In this section, we introduce the algorithms and shortly explain the concepts from other authors which we implement and compare for TDE. From now on, for the sake of easier understanding, we focus on the current of the electric machine (EM) i_{EM} and the HVB i_{HVB} (without other consumers than the EM) as examples. Nevertheless, the proposed methods can be extended to every current signal in the HV system of an EV. Furthermore, we inverse the sign of i_{HVB} from now on to make its shape similar to the one of the EM. Thus, we can treat the HVB current signal as delayed or preceded version of the EM respectively.

Our goal is to find the time delay t_d in a bus system which can be described as

$$\begin{aligned} x_1(t) &= i_1(t) + n_1(t) \\ x_2(t) &= i_2(t - t_d) + n_2(t - t_d), \end{aligned} \tag{1}$$

where t stands for the time step, $x_1(t)$ is the measurement signal of the faster component, $x_2(t)$ describes the slower component's signal, $i_1(t)$ and $i_2(t)$ describe the corresponding currents and $n_1(t)$ and $n_2(t)$ are noise terms [4]. As we cannot retrieve the currents $i_1(t)$ and $i_2(t)$ directly, we cannot minimize the difference between $i_1(t)$ and $i_2(t)$. Instead, we directly minimize the difference between the two measurement signals $x_1(t)$ and $x_2(t)$.

A. Linear Regression

Our first approach is to use Linear Regression to identify the *basis functions* of two received signals and compare the horizontal offset between these functions. As degree of the basis function, we choose a parabola for two reasons. First, the sampling frequency of our measurements is high enough to fit the signals with a parabola for a short time duration. Second, the comparison of the horizontal offset is easiest with a parabola because it only has one extremum.

We collect the last M measurements x_k of the HV components $k \in \{\text{EM}, \text{HVB}\}$ in a measurement vector $\boldsymbol{y_k} = \begin{pmatrix} x_k(t) & x_k(t-1) & \cdots & x_k(t-M+1) \end{pmatrix}$. Then, we retrieve the weight vector $\boldsymbol{w_k} = \begin{pmatrix} w_{k,0} & w_{k,1} & w_{k,2} \end{pmatrix}^T$ with Linear Regression [5] according to

$$\boldsymbol{w_k} = \left(\sum_{n=1}^{N} \boldsymbol{\phi}^n (\boldsymbol{\phi}^n)^T \right)^{-1} \sum_{n=1}^{N} y_k^n \boldsymbol{\phi}^n. \tag{2}$$

Here, the notation y_k^n and $\boldsymbol{\phi}^n$ represents the n-th column of $\boldsymbol{y}_k$ and $\boldsymbol{\phi}$, respectively. The so-called *design matrix*

$$\boldsymbol{\phi} = \begin{pmatrix} 1 & t & t^2 \\ 1 & t-1 & (t-1)^2 \\ \vdots & \vdots & \vdots \\ 1 & t-M+1 & (t-M+1)^2 \end{pmatrix}^T$$

consists of $N = 3$ columns in our case.

With the weight vector from (2), we are able to fit a parabola

$$f_k(t) = w_{k,0} + w_{k,1}t + w_{k,2}t^2 \tag{3}$$

as basis function to the measurement vector $\boldsymbol{y_k}$.

After retrieving the basis functions in (3), we transfer them into vertex form

$$f_k(t) = w_{k,2}(t - x_{k,\text{vertex}})^2 + y_{k,\text{vertex}} \tag{4}$$

to identify the coordinates $(x_{k,\text{vertex}}, y_{k,\text{vertex}})$ of each basis function's vertex. The time delay between the EM and the HVB current signals is then given by the difference on the time axis between their vertices according to

$$t_d = x_{\text{EM, vertex}} - x_{\text{HVB,vertex}}. \tag{5}$$

B. Variance Minimization

Our second approach is to minimize the variance of the difference between two signals $x_1(t)$ and $x_2(t)$ by shifting the signal $x_2(t)$ forward.

Like in III-A, we collect the two signals $x_1(t)$ and $x_2(t)$ for M time steps. A straightforward idea for the minimization of the difference between $x_1(t)$ and $x_2(t)$ is to minimize their estimated mean squared error (MSE)

$$\text{MSE}(x_1(t), x_2(t)) = \frac{1}{M - T_{\text{max}}} \sum_{t=1}^{M-T_{\text{max}}} (x_1(t) - x_2(t+t_{d,i}))^2 \tag{6}$$

with different time shifts $t_{d,i}$ in a pre-defined range $t_{d,i} \in [T_{\text{min}}, T_{\text{max}}]$ with $T_{\text{max}} < M$. However, our experiments show that we need a relatively high M to achieve stable results. We can significantly minimize M, if we take the estimated expected value

$$E = \frac{1}{M - T_{\text{max}}} \sum_{t=1}^{M-T_{\text{max}}} (x_1(t) - x_2(t + t_{d,i})). \tag{7}$$

into account. Thus, instead of minimizing the MSE from (6), we minimize the estimated variance of the difference between the two signals

$$\sigma^2 = \frac{1}{M - T_{\text{max}}} \sum_{t=1}^{M-T_{\text{max}}} ((x_1(t) - x_2(t+t_{d,i})) - E)^2. \tag{8}$$

The time delay between the EM and the HVB current signals is the $t_{d,i}$ that minimizes the variance

$$t_d = \underset{t_{d,i}}{\text{argmin}}\ \sigma^2. \tag{9}$$

As we do not know in the beginning whether $x_{\text{EM}}(t)$ or $x_{\text{HVB}}(t)$ is the faster signal, we have to choose one of them as $x_1(t)$ and the other one as $x_2(t)$ for the first execution and try $T_{\text{min}} = -T_{\text{max}}$. From the second execution on, the value of T_{min} and T_{max} can be reduced and chosen recursively, because the EV's bus system usually changes its time delay only once in the beginning, but not during advanced execution. Therefore, we choose $T_{\text{min}}(t) = t_d(t-1) - 1$ and $T_{\text{max}}(t) = t_d(t-1) + 1$ from the algorithm's second execution on.

C. Adaptive Filter

Emadzadeh et al. model the time delay as finite impulse response (FIR) filter. To find the optimal filter parameters, they use Least Mean-Squares (LMS) and other related algorithms. Their approach results in a similar one which we use in III-B with the difference that they minimize the MSE instead of the variance. For further details, we kindly refer the reader to [4].

IV. Experimental Setup

In this section, we explain the data and the setup for the experiments to evaluate the performance of the three concepts for TDE.

A. Data

For the evaluation of the three concepts shown in III, we use 74 data sets. The data sets contain all currents of the HV system and are recorded during representative drives on public roads with close to production EVs. The recordings correspond to $10\,\text{h}\ 33\,\text{min}$ of driving. For the experiments, the 74 data sets are divided into 409 sub-data sets with a maximum length of 10,000 time steps. The minimum length among the 409 sub-data sets is 1,807 time steps.

B. Experiments

According to *Kirchhoff's current law*, we assume that the sum of the measurements of i_{HVB}, i_{EM}, i_{heat}, i_{cool} and i_{DCDC} is zero. Thus, we estimate the *ground truth* of the time delay for each data set by minimizing the MSE of the complete data set (see equation 6). In this case, M is the length of the data set, T_{max} is 10 time steps and T_{min} is -10 time steps, since the time delay in the EV is normally smaller than ten time steps.

For the evaluation of the variance minimization concept we test different frame sizes $M \in \{30, 50, 100, 200, 300\}$. In the first calculation, we also choose $T_{\text{max}} = 10$ and $T_{\text{min}} = -10$. From the second execution on we select $T_{\text{min}}(t) = t_d(t-1) - 1$ and $T_{\text{max}}(t) = t_d(t-1) + 1$.

In the experiment of the Adaptive Filter concept, we choose the frame size of 2^{10} as proposed by [4]. Additionally, we execute our experiments with a more efficient frame size of 2^8. We assume the length of the adaptive filter to be 10. All the other parameters for the used

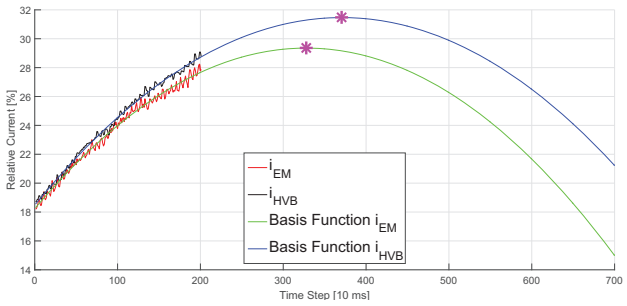


Figure 3. Basis functions of i_{HVB} (black) and i_{EM} (red) simulated by Linear Regression (blue and green, respectively). The magenta marked points are the vertexes. Their horizontal difference is 43 time steps in contrast to the real time delay which is 6 time steps. The wrong TDE is caused by the noise and the vertical offset of the measurement signals.

algorithms are chosen equally to [4]. Furthermore, we evaluate five different learning rates for LMS.

For each of the three concepts, we calculate the time delay every 20th time step. In total, this results in around 90,000 time delay estimations for each concept.

C. Environment

The three concepts are implemented in Matlab R2015b with Microsoft Windows 10 on an HP® EliteBook™840 G3 with an Intel® Core™i5-6300U 2.40GHz CPU and 8 GB RAM.

V. Results

In this section, we present the results of our experiments and evaluate the performance of the three concepts.

A. Linear Regression

The first approach Linear Regression is, to a large extent, affected by noise and the offset between the two signals caused by measurement inaccuracies. Especially this offset leads to an imprecise estimation of the vertices and thus a wrong estimated time delay. Figure 3 shows an example for such a wrong estimation. In this data set, the time delay between i_{HVB} and i_{EM} is equal to 6 time steps. We train both curves on 200 measurement samples of their corresponding signal. However, due to noise and some vertical offset between the signals the vertex of the slower signal is not only shifted to the right but also to the top. The shift in vertical direction also affects the horizontal position of the vertex and results in a wrong TDE of 43 time steps.

Table I shows the results and the average run-time of this approach with three different frame sizes. The run-time grows with increasing frame sizes, whereas the RMSE becomes smaller. Nevertheless, the RMSEs are in general very high even for large frames.

B. Variance Minimization

Table II shows the RMSE between the ground truth of the time delay and the calculated time delay as well as the

Table I. RMSE and Run-Time Analysis of the Linear Regression Concept.

	RMSE	Average Run-Time (s)
Frame Size 30	$5.83 \cdot 10^{10}$	$1.74 \cdot 10^{-4}$
Frame Size 200	$4.92 \cdot 10^{4}$	$9.80 \cdot 10^{-4}$
Frame Size 300	$4.47 \cdot 10^{4}$	$1.40 \cdot 10^{-3}$

Table II. RMSE and Run-Time Analysis of the Variance Minimization Concept.

	RMSE	Average Run-Time (s)
Frame Size 30	2.0696	$4.54 \cdot 10^{-5}$
Frame Size 50	1.3034	$4.70 \cdot 10^{-5}$
Frame Size 100	1.2364	$4.77 \cdot 10^{-5}$
Frame Size 200	1.2215	$5.16 \cdot 10^{-5}$
Frame Size 300	1.1825	$5.79 \cdot 10^{-5}$

average of the run-time for each time delay calculation, corresponding to different frame sizes M (in equation 8). We can see that the concept requires a relatively short run-time as it benefits from the recursive calculation only in the area $t_{d,i} \in [T_{\text{min}}, T_{\text{max}}]$ with $T_{\text{min}}(t) = t_d(t-1)-1$ and $T_{\text{max}}(t) = t_d(t-1) + 1$. In addition, the RMSE decreases while the size of the frame increases. The accuracy has a large improvement when the frame is enlarged from 30 time steps to 50 time steps.

C. Adaptive Filter

Based on the learning rate and parameters in [4], the Recursive Least-Squares (RLS) algorithm performs better than the LMS, Normalized Least Mean-Squares (NLMS) and Least Mean-Fourth (LMF) algorithms (see table III).

Furthermore, we analyze the learning rate for the LMS algorithm. As mentioned in [4], the learning rate μ is typically chosen in the range $0 < \mu < 2/(M\sigma_u^2)$, where σ_u^2 is the input signal variance and M is the length of the filter. Thus, we compare the performance of LMS with different $\mu = a/(M\sigma_u^2)$ and $a \in \{0.01, 0.05, 0.1, 0.5, 1\}$.

Table III. RMSE and Run-Time Analysis of the Adaptive Filter Concept for Different Algorithms with a Frame Size of 2^{10}.

	RMSE	Average Run-Time (s)
LMS	2.8972	$1.13 \cdot 10^{-2}$
NLMS	2.5163	$1.31 \cdot 10^{-2}$
LMF	2.6138	$1.14 \cdot 10^{-2}$
RLS	2.276	$1.07 \cdot 10^{-2}$

Table IV. RMSE AND RUN-TIME ANALYSIS OF LMS WITH DIFFERENT LEARNING RATES AND A FRAME SIZE OF 2^{10}.

	RMSE	**Average Run-Time (s)**
$a = 0.01$	3.0451	$1.14 \cdot 10^{-2}$
$a = 0.05$	2.3608	$1.14 \cdot 10^{-2}$
$a = 0.1$	2.1479	$1.14 \cdot 10^{-2}$
$a = 0.5$	2.6484	$1.14 \cdot 10^{-2}$
$a = 1$	3.2474	$1.14 \cdot 10^{-2}$

Table V. RMSE AND RUN-TIME ANALYSIS OF THE LMS ALGORITHM WITH A SHORTER FRAME SIZE OF 2^8.

	RMSE	**Average Run-Time (s)**
$a = 0.01$	2.8603	$2.80 \cdot 10^{-3}$
$a = 0.05$	2.5598	$2.80 \cdot 10^{-3}$
$a = 0.1$	2.3994	$2.80 \cdot 10^{-3}$
$a = 0.5$	2.4983	$2.80 \cdot 10^{-3}$
$a = 1$	3.0781	$2.80 \cdot 10^{-3}$

In table IV, the RMSE has the minimal estimation RMSE of 2.1479 with $a = 0.1$. It is much smaller than the RMSE of 2.8972 with the fixed learning rate in [4]. For a too large or a too small a, the performance of the LMS decreases significantly. This result is expected, since a too small learning rate leads to slow convergence while a large learning rate most often misses the optimum.

In addition, we evaluated the LMS algorithm with a more efficient frame size of 2^8. As printed in Table V, the smaller frame size improves the run-time. Although some non-optimal learning rates improve their estimation accuracy, which we explain with the drop of local minima due to the shortened frame, the two best learning rates in the experiment with the frame size of 2^{10} increase their estimation errors with the smaller frame size.

VI. DISCUSSION

We discuss the advantages and drawbacks of the previously described and evaluated concepts in this section.

Linear Regression has the advantage that it can directly find out the faster component. Thus, one single execution during the same time step for the same signal is sufficient even in the beginning, which makes it interesting, if a computational effective approach is needed. However, its efficiency suffers from the matrix inversion in (2). Even worse, it is least accurate of the three proposed concepts due to noise and vertical offsets between the signals. The high estimation errors make this approach infeasible for our purpose. Another drawback is that a parabola is not always the optimal basis function for the regression of measurement signals.

The Variance Minimization approach does not require such a basis function. Unfortunately, it is not able to detect the faster signal without trying each possible time delay for both signals. This results in a computational expensive brute force calculation in the first time step. Afterwards, it is very efficient because it only must execute basic math operations and searches only for a restricted number of possible delays. Compared to the other approaches, Variance Minimization requires the smallest frame size to retrieve feasible results. In total, Variance Minimization is most accurate and efficient among the proposed approaches.

Although it is not as efficient and accurate as Variance Minimization, the Adaptive Filter approach still retrieves better results than Linear Regression. The best results for Adaptive Filters, in our case, are reached with the LMS algorithm and a learning rate of $\mu = 0.1/(M\sigma_u^2)$ (see Table IV). The learning rate, which must be chosen manually, is the worst drawback of this algorithm. It can lead to sub-optimal learning, if the user chooses a wrong value. In contrast to LMS, the RLS algorithm does not require a learning rate. However, we see that the RLS algorithm has lower accuracy and longer running time than the variance minimization concept (see Tables II and III).

VII. CONCLUSION AND OUTLOOK

This paper presents three different approaches for TDE of measurement signals in the power train of EVs. As automotive ECUs are designed very efficiently, our evaluation's focus lies also on computation complexity and not solely on accuracy. Unfortunately, Linear Regression is not suited for our purposes because it suffers too much from vertical offsets in the measurement data. But with Variance Minimization, we present a feasible approach for TDE of distributed sensor systems of EVs. We recommend to use Variance Minimization due to its high estimation accuracy and computational efficiency. Adaptive Filters are also not suited because they require too large frame sizes and have lower accuracy than Variance Minimization.

After the introduction of an automated TDE system, we now know each signal's delay. However, if we correct the delay, we move some signals to the past and lose the measurements corresponding to the latest time steps. This is correct because in fact we do not receive up-to-date measurements, only delayed ones from the past. We really do miss the last measurements and there is a gap between the last received measurement and the present time step. Thus, our next work focuses on possible ways to close this gap by replacing the hidden information about the missing measurements from the latest time steps.

REFERENCES

[1] P. Komarnicki, J. Haubrock, and Z. A. Styczynski, "Electrical components of an EV (Elektrische Komponenten des E-Kfz)," in *Elektromobilität und Sektorenkopplung*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2018, pp. 61–109.

[2] H. Zeng, M. Di Natale, P. Giusto, and A. Sangiovanni-Vincentelli, "Statistical analysis of Controller Area Network message response times," in *2009 IEEE International Symposium on Industrial Embedded Systems*. Lausanne, Switzerland: IEEE, Jul. 2009, pp. 1–10. [Online]. Available: http://ieeexplore.ieee.org/document/5196185/

[3] L. Svilainis, A. Aleksandrovas, K. Lukoseviciute, and V. Eidukynas, "Investigation of the Time of Flight estimation errors induced by neighboring signals," in *2013 IEEE 7th International Conference on Intelligent Data Acquisition and Advanced Computing Systems (IDAACS)*. Berlin, Germany: IEEE, Sep. 2013, pp. 413–418. [Online]. Available: http://ieeexplore.ieee.org/document/6662718/

[4] A. A. Emadzadeh, C. G. Lopes, and J. L. Speyer, "Online time delay estimation of pulsar signals for relative navigation using adaptive filters," in *2008 IEEE/ION Position, Location and Navigation Symposium*. Monterey, CA, USA: IEEE, 2008, pp. 714–719. [Online]. Available: http://ieeexplore.ieee.org/document/4570029/

[5] D. Barber, *Bayesian Reasoning and Machine Learning*. Cambridge: Cambridge University Press, 2011. [Online]. Available: http://ebooks.cambridge.org/ref/id/CBO9780511804779

The next paper is an extended version of the previous one. It has been published in the scope of a special issue for selected papers of the 10th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS). Besides an extended introduction and more extensive research into the state of the art, the extended paper offers a new, optimized version of the Variance Minimization approach. The optimization consists of a quantification of the reliability of new input data. If the reliability is not sufficiently large, we do not update the estimated time delay. This way, our resulting estimated time delay between two measurement signals becomes more reliable.

Hypotheses H1 and H2a are not affected compared to the previous paper. But Hypothesis H3 is validated even more by the additionally proposed solution. However, regarding Hypothesis H4, the workload for the applicators unfortunately grows here. In this case, it is necessary to balance the stability of the results with the workload for the applicators.

Since the following paper is an extension of the previous one, my contributions here are the same. Additionally, I am responsible for the extended sections dealing with the introduction and the state of the art. The development of the new, optimized Variance Minimization approach is Xuyi Wu's and Ahmed Ayadi's achievement. Furthermore, Ahmed Ayadi is responsible for the subsections explaining and evaluating the method. Like in the previous version of the paper, it is my task to review all changes made by the authors as well as to integrate the external reviewers' feedback to the paper. Additionally, it is my responsibility to acquire the funding for the publication from the BMW Group.

sensors

MDPI

Article

Evaluation of Three Different Approaches for Automated Time Delay Estimation for Distributed Sensor Systems of Electric Vehicles

Jakob Pfeiffer [1,2,*], Xuyi Wu [2] and Ahmed Ayadi [2]

[1] BMW Group, Petuelring 130, 80788 Munich, Germany
[2] Department of Electrical and Computer Engineering, Technical University of Munich, Arcisstr. 21, 80333 Munich, Germany; Xuyi.Wu@tum.de (X.W.); Ahmed.Ayadi@tum.de (A.A.)
* Correspondence: Jakob.J.Pfeiffer@bmwgroup.com

Received: 3 December 2019; Accepted: 26 December 2019; Published: 8 January 2020

Abstract: Deviations between High Voltage (HV) current measurements and the corresponding real values provoke serious problems in the power trains of Electric Vehicles (EVs). Examples for these problems have inaccurate performance coordinations and unnecessary power limitations during driving or charging. The main reason for the deviations are time delays. By correcting these delays with accurate Time Delay Estimation (TDE), our data shows that we can reduce the measurement deviations from 25% of the maximum current to below 5%. In this paper, we present three different approaches for TDE. We evaluate all approaches with real data from power trains of EVs. To enable an execution on automotive Electronic Control Units (ECUs), the focus of our evaluation lies not only on the accuracy of the TDE, but also on the computational efficiency. The proposed Linear Regression (LR) approach suffers even from small noise and offsets in the measurement data and is unsuited for our purpose. A better alternative is the Variance Minimization (VM) approach. It is not only more noise-resistant but also very efficient after the first execution. Another interesting approach are Adaptive Filters (AFs), introduced by Emadzadeh et al. Unfortunately, AFs do not reach the accuracy and efficiency of VM in our experiments. Thus, we recommend VM for TDE of HV current signals in the power train of EVs and present an additional optimization to enable its execution on ECUs.

Keywords: automotive; current; electric power train; electric vehicle; embedded systems; delay; detection; distributed systems; measurements; power train; sensor; signals; time delay estimation

1. Introduction

Political guidelines in various countries to decarbonize individual mobility led to an exponential growth of Electric Vehicles (EVs) in offers and sales. However, one obstacle for the success of EVs is the so-called range anxiety [1]. Customers are afraid that an EV is not able to provide the range they need for all of their journeys. To combat range anxiety and increase the range of EVs, there are two different ways. The first one is to simply increase the size of the High Voltage Battery (HVB). Unfortunately, this means to increase the size of the most expensive component of an EV, and after all, it is not a very sustainable way. The second way, which is our solution of choice, is to make EVs more efficient.

Kirchhoff's current law states that the sum of all currents at a node of an electric system is equal to 0 A. However, considering measurement signals of nodes in the power trains of EVs with distributed sensor systems, the sum of all currents can differ up to 20 % of the maximum current (see Figure 1). If we look closer at the Root Mean Square Error (RMSE) of the sum of currents $\mathrm{RMSE}(i_{\mathrm{sum}}) = 0.67\%$, we realize that it has the same value as the mean current of the DCDC converter $\mu_{i_{\mathrm{DCDC}}} = 0.67\%$.

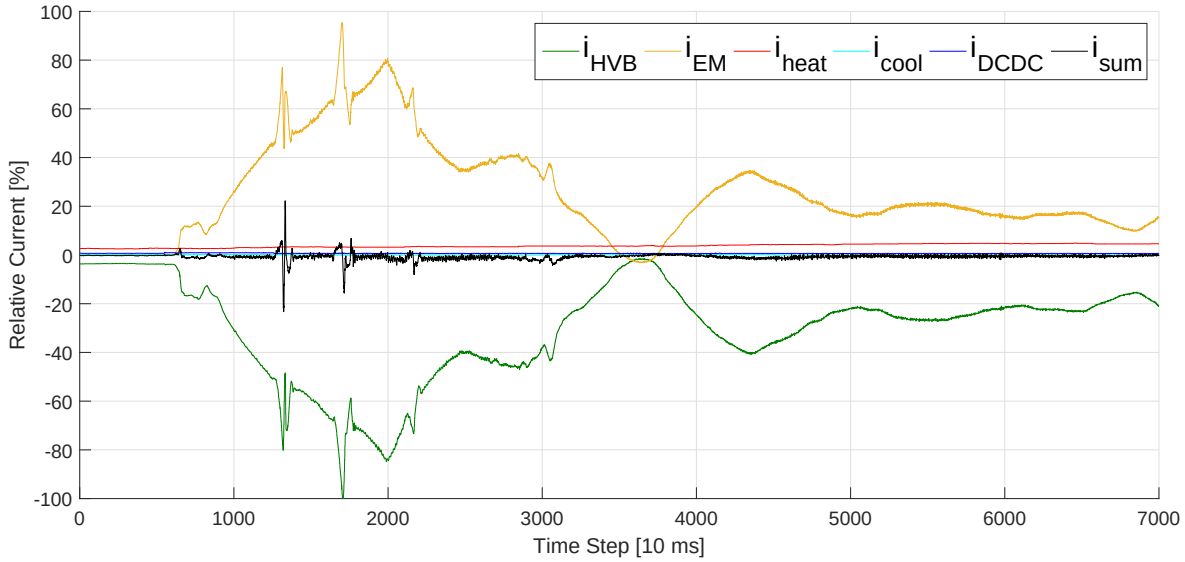


Figure 1. Currents of all HV components in an EV on a test drive. The sum of all currents i_{sum} is plotted in black. According to *Kirchhoff's current law*, it should be constantly 0 %. However, looking at the measurements shows that the deviation i_{sum} is higher than the current of the DCDC converter i_{DCDC}. Even its noise spectrum is approximately half as high as the consumption of the heating i_{heat}, which is the second largest consumer in this drive.

A different value than 0 A for the sum of all currents indicates that there is a divergence between measurements and real values. The divergence becomes problematic when the power train is operating close to the system boundaries. For example, there are boundaries for the protection of the HVB. The HVB is only capable of discharging or charging a restricted amount of power. Higher amounts would threaten the HVB's lifetime and safety [2]. To ensure a safe operation mode even for high divergences between measurements and real values, additional protection offsets (see Figure 2) might be added to the boundaries, although they have some drawbacks.

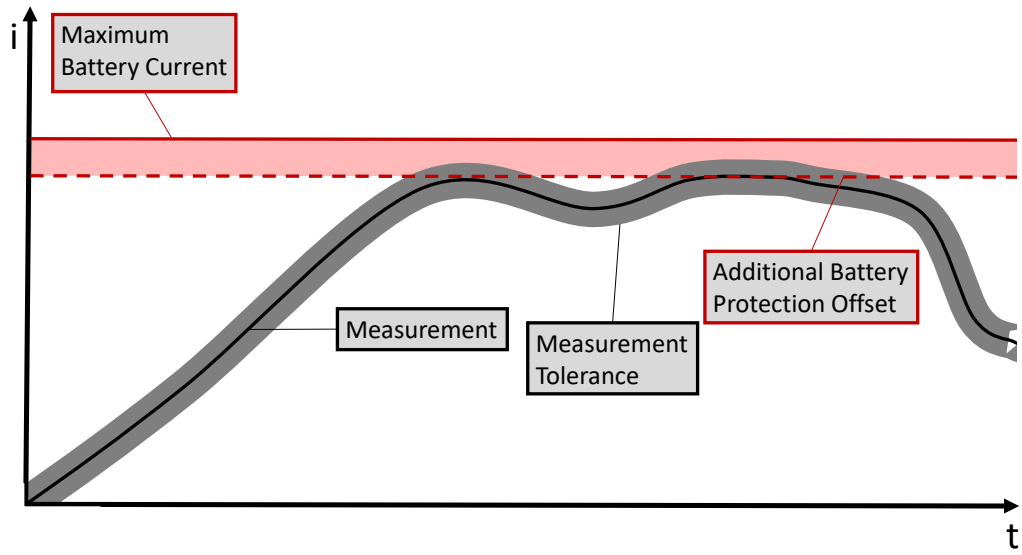


Figure 2. A simplified example of offsets for protection of the HVB. The measured value (black) differs from the real value in the range of some tolerance (grey). To prevent exceeding the battery limit (red, solid) even under the worst measurement conditions, an additional battery protection offset (red, dashed) is introduced. The same principle is used analogously with negative currents. It can be extended to other HV components.

For example, in the charging case, most notably during recuperation, the HVB might not allow the full power level, even though it would be capable of handling it. Thus, the amount of power charged to the battery is restricted and the EV loses cruising range while its power consumption increases. In the opposite case, the system might not release requested power, although the HVB could provide it in reality. This additional restriction of power decreases the EV's performance. As can be seen from the two examples above, minimizing the magnitude of the protection offsets also allows increasing the performance as the efficiency and the cruising range of EVs.

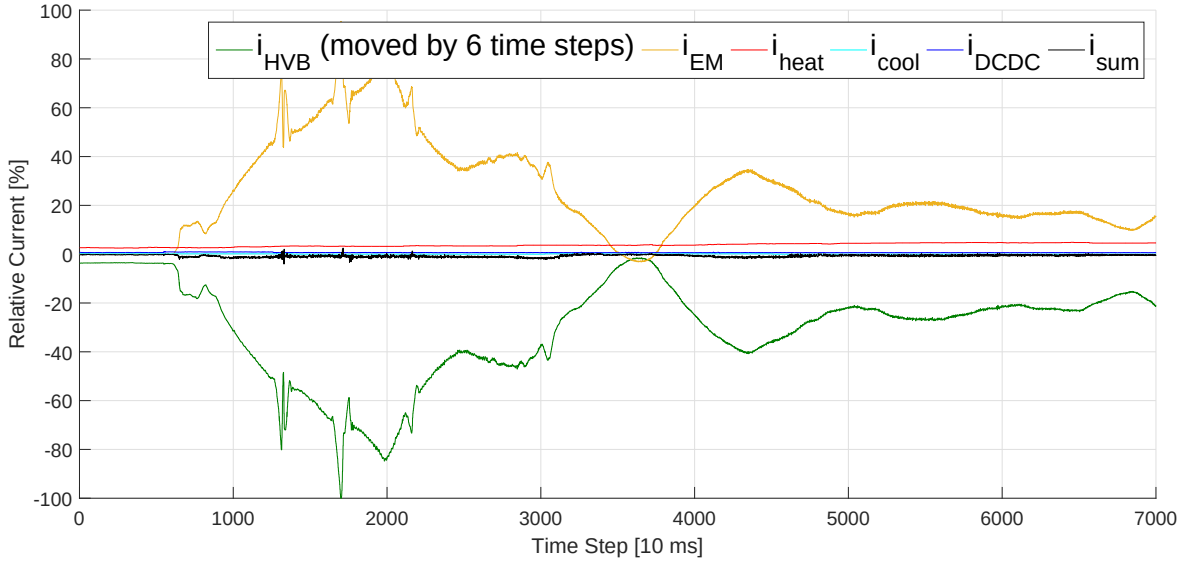


Figure 3. The same test drive as in Figure 1 but with the battery current i_{HVB} (green) shifted by six time steps. The sum of all currents i_{sum} (black) is significantly closer to 0 %.

Besides measurement faults and sensor uncertainties [3], the divergence between measurements and real values is caused by time delays. Figure 3 shows an example of the sum of all currents i_{sum} being reduced by shifting a signal by 6 time steps. The delays result from distributed sensor systems in the power train as plotted in Figure 4. The High Voltage (HV) components have their own Electronic Control Unit (ECU) which is connected with the current sensors and processes the sensor information. The ECUs exchange this information via bus systems. The buses require individual amounts of time to send the measurement signals. Thus, from an ECU's point of view, the sensor information from other ECUs arrives with individual delays (see the Ego ECU in Figure 4). These individual delays could be compensated easily with a synchronized clock and time stamps as part of each bus message. However, this solution would have two drawbacks. First, it would increase the bus traffic as not only the measurement information must be carried by the messages but also the time stamp. As a result, the EV would either require a faster bus which is able to transport more information, or it would have to reduce the information exchanged between the ECUs. Second, there exists no clock in the power trains of modern series EVs which is synchronized with all ECUs at the same frequency as the message exchange. Usually, the ECUs are synchronized in a longer time frame than they communicate. Thus, the time stamp solution would require additional or higher performing hardware and increase the costs for the production of the EV.

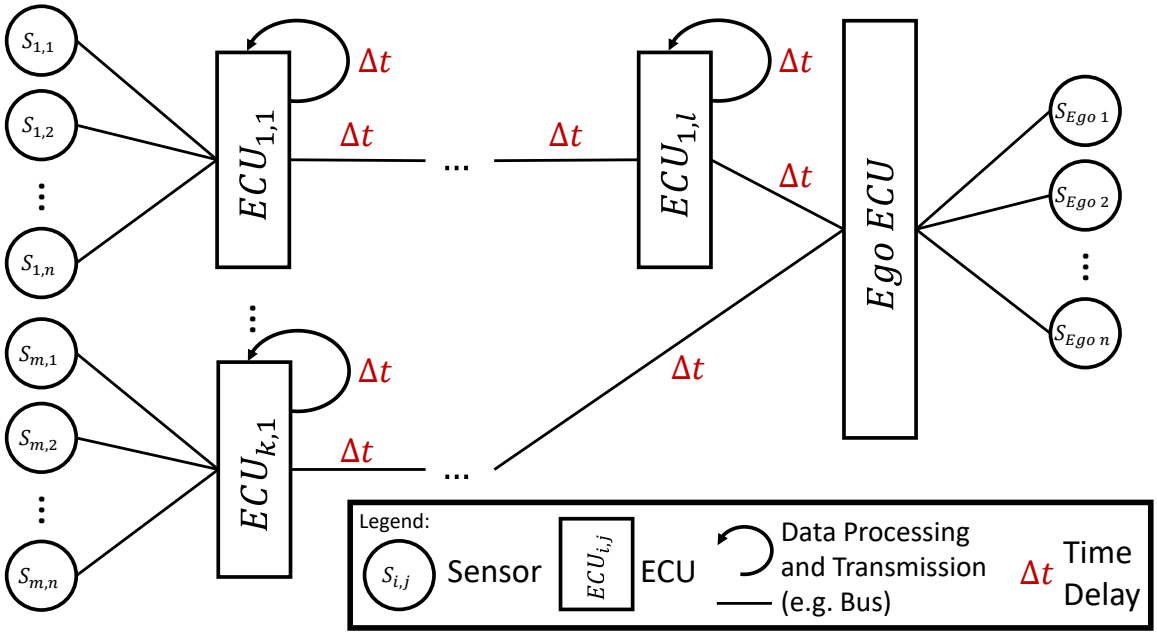


Figure 4. A schematic example of an automotive bus system with higlighted sources of time delays. Please note that the time delays are highly individual and not necessarily equal, but constant or only slowly changing. The ECUs can be connected directly or indirectly via other ECUs. The Ego ECU is not able to reconstruct the time delays, because it only knows the received measurement values and their last sender. It has no further information about the time passed since the measurement's original creation.

The aim of this work is to automatically detect the time delay between measurement signals from different sensors without additional hardware. For this purpose, we develop two different approaches. One of them is based on Linear Regression (LR), whereas the other one optimizes the estimated variance of the difference between several signals. We compare our approaches to other state-of-the-art Time Delay Estimation (TDE) algorithms and evaluate them with a focus on precision and run-time efficiency. Apart from allowing a more accurate power distribution, the automated TDE helps to reduce the battery protection offset and thus to increase the performance, efficiency and cruising range of EVs.

The rest of this paper is structured as follows. Section 2 states related work and the similarities and differences to our work. Furthermore, Section 2 highlights the contributions of our work to the state of the art. In Section 3, we explain the theory behind our work before we describe the practical experiments in Section 4. The experiments' results, stated in Section 5, show us the performance of the algorithms for our use case. Based on this evaluation, we take the best performing algorithm and optimize it further. The optimization steps can be taken from Section 3.4 and their impacts to the results from Section 5.4. In Section 6, we discuss the advantages and drawbacks of all proposed concepts. Finally, we draw our conclusions and give a short outlook in Section 7.

2. State of the Art

There exists plenty of literature about TDE, although—to the best of our knowledge—none of them is tailored to the specific problem of TDE of current signals in EVs. In the following, we present several publications about TDE from different fields of application, such as embedded systems, acoustics, medicine, positioning, aeronautics, process technology, and robotics.

An approach which also deals with EVs and time delays is the one by Guo et al. [4]. However, their approach is similar to ours only at the first look. Their goal is to stabilize a grid of electric sources and sinks with EVs. For the stabilization of the grid, they propose time delay resistent control strategies

of smart grids with EVs. The EVs are able to charge bidirectionally. The bidirectional charging is used to smooth disturbances and respond rapidly to fast occurring changes in the power distribution of the grid. An example for such a rapidly occurring change in the times of renewable energies is the power output of wind turbines when a strong wind occurs. Compared to our approach, Guo's focus is rather on the control strategy than on the TDE. Another difference with our work is that Guo's system is rather macroscopic with lots of different elements and many EVs in the grid. Our system is instead quite microscopic. We consider a single EV with a power train of around five sources and sinks. Our communication network might be smaller than the number of HV components as some consumers might share the same ECU. For example, the heating and the cooling component of an EV use both the climate control ECU for bus communication.

Kali et al. [5] propose a controller with TDE for Electric Machines (EMs). The TDE is executed state-based with the help of a model of the EM. The model design demands expert knowledge about the physical principles of an EM. This is justified for Kali et al. as they require the same knowledge for their controller. However, for our case, we want to be able to estimate the time delays without further knowledge about the HV components. Our TDE shall be executable with nothing else than the available measurement data.

Zeng et al. [6] introduce a statistical approach to predict the delay of a bus message. The content of the messages does not need to be known to achieve high accuracy. This is different from our scenario where we want to make use of the information carried by the message. In contrast to Zeng et al., we do not require predicting the time delay accurately to milliseconds. For our purposes, an estimation of the number of delayed discrete time steps is sufficient.

Not from the field of electric power trains or bus communication, but from acoustics is the approach shown by Lourtie and Moura [7]. They use a stochastic approach to model time delays in an acoustic path environment. Like ours, their environment consists of several sources. However, in contrast to our scenario, the delay they want to estimate varies with time. In our case, we assume the time delay to be constant in a short time frame. For longer periods, it might change slowly. The reason for the slowly changing time delay is that it is caused during the wake up procedure of the EV. The ECUs wake up in an unsynchronized way. Afterwards, the ECUs are synchronized on a relatively large time frame (e.g., 1 s), but work based on short time steps (e.g., 10 ms).

Another acoustics application for TDE is shown by He et alii [8]. They use the so-called Multichannel Cross-Correlation Coefficient algorithm to estimate time delays of speech sources in noisy and reverberant environments.

Svilainis et al. [9] present another interesting approach. Their goal is to estimate the time passed between emitting an ultrasonic signal and absorbing its reflection. Like Zeng et al., they require high precision. Another difference to our approach is that their algorithms make use of the pulse form of ultrasonic signals. Our signal as plotted in Figure 1 can vary in a large range and does not necessarily contain pulses (e.g., after time step 5,000).

Mirzaei et al. expand TDE for ultrasonic signals to the field of medicine [10]. The authors introduce a window-based TDE approach to estimate the time passed between two frames of radio-frequency data. They compare the results of the new window-based approach to their previously developed, optimization-based method [11] and to Normalized Cross-Correlation.

Recently, Garcez et al. published their work on a similar problem to ours, but in a completely different field of application [12]. Like bus systems of EVs, Global Navigation Satellite Systemss (GNSSs) systems have real-time requirements. Their goal is to minimize deviations between measurements and real position data. The time delays are caused during the transmission of GNSS messages, when the signals do not take straight lines of sight, but are reflected on their way or suffer from noise. The authors propose a tensor-based subspace tracking algorithm to efficiently estimate time delays of received GNSS signals.

A similar approach is presented by Xie et al. for an indoor positioning sensing system [13]. They sense positions of mobile devices based on the signal strength and the signal's time delay since

its transmission from a base station. For the TDE, Xie et al. combine Cross-Correlation with Quadratic Fitting. This is similar to our LR approach (see Section 3.2), where we try to fit the signals with quadratic functions to retrieve the delay between them. Like Garcez et al., they have to deal with the problem that the signals are often reflected and do not take direct lines of sight. Different to Garcez et al., Xie's approach takes the strength of the signal into account for retrieving a more exact position estimation. For our work, we cannot take advantage of this information, because in wire-based bus systems all signals are equally strong.

Schmidhammer et al. estimate positions of moving, non-cooperative objects in vehicular environments [14]. Their idea is to estimate the position of an object based on time delays in a network of distributed receiving and transmitting nodes. In contrast to our work, the networking nodes of Schmidhammer et al. are not necessarily on-board the vehicle, but can also be mounted on the road infrastructure.

Emadzadeh et al. [15] show an inspiring approach for detecting the relative position of spacecrafts. For retrieving the position, they examine an X-ray signal received by two spacecrafts and determine the time delay between them. For the TDE, they use Adaptive Filters (AFs). This approach seems very promising to us. We implement the algorithms of Emadzadeh et al. and compare them to ours in order to find out if their approach can be transferred from X-ray signals to current measurements in the power train of EVs.

Like Emadzadeh et al., Liu et al. focus on AFs [16]. Compared to our problem of fixed or only slowly changing time delays, the difference in Liu et al. is that they deal with time-varying time delays. That makes further processing steps necessary. For example, they require a transition probability matrix and an initial probability distribution vector to model the time delay changes with a Markov chain.

Park et al. analyze time series data with Autoencoders and Long Short-Term Memory Neural Networks (LSTMs) to detect faults in industrial processes [17]. The authors emphasize the importance of TDE for correct fault detection. However, they focus only on time delays caused by their own fault detection system. Our focus lies on earlier steps in the processing chain. We want to detect time delays between the input signals before they are passed to other computation processes. Furthermore, we want to implement algorithms which are able to learn on-board the automotive ECUs and adapt themselves to new data. As the training of Neural Networks is quite memory intensive and demands high computational power, they do not belong to our methods of choice.

Close to the application field of industrial processes is the approach of Srinivasa Rao et al. [18]. In their recent article, the authors propose fuzzy parametric uncertainty to mathematically model systems with time delays. Their goal is to enable a robust controller design. For this purpose, they first approximate the time delay system as an interval system. After retrieving the intervals, they design an optimal controller for these. Like Guo et al., Srinivasa Rao et al. focus on how to retrieve an optimal controller, which is not part of our work. Although they focus on the control of industrial processes, their article is very general. Besides industrial plants, they also mention potential fields of application, such as EMs or robot manipulators.

Time delay compensation for robots is the focus of Shen et al [19]. Their focus is on teleoperating robots which require knowledge about the time delay between the master and the slave robot for stable operation. The robots and their communication channels are modeled as extended dynamical system. For this system, Shen et al. develop a cascade observer which is able to control it in a stable way. The authors assume that a sufficiently accurate value for the TDE is given and concentrate on its compensation. This is different to our work here. We explicitly want to estimate the time delay.

You et al. develop a proportional multiple integral observer for fuzzy systems [20]. The goal of their work is the same as ours. They want to minimize deviations between measurements and real values caused by time delays and measurement inaccuracies. Their time delays are also varying. Unlike the varying time delays presented before, the ones of You et al. do not vary with time but rather with states. Their focus is also on industrial processes and not on electric power trains. However, the

main difference between our works is that You et al. want to minimize time delays and measurement inaccuracies with the same system.

Our approach follows the *divide and conquer* strategy and faces the two problems separately. We focus on the problem of measurement deviations caused by measurement inaccuracies in our previous work [3]. However, measurement inaccuracies are not part of this work. Here, we assume that the measurements are appropriately accurate and that the main deviations are caused by time delays as shown in Figure 1 and Figure 3. Thus, TDE is our solution of choice to minimize the deviations.

Our contribution in this article is the development of a regression-based approach and an algorithm based on Variance Minimization (VM) for TDE as first presented in [21]. We transfer the ideas introduced by Emadzadeh et al. to the domain of currents in the HV system of EVs and compare the results to our approaches in matters of accuracy and computational performance. Our TDE works only with the data available in modern series EVs and does not require an additional clock. In addition to [21], we introduce an optimization of the most accurate and efficient of our evaluated approaches. We further evaluate the optimization both on artificially created data with known ground truth as well as real drive data with unknown ground truth.

3. Concepts

In this section, we introduce the algorithms and shortly explain the concepts from other authors which we implement and compare for TDE. From now on, for the sake of easier understanding, we focus on the current of the EM i_{EM} and the HVB i_{HVB} (without other consumers than the EM) as examples. Nevertheless, the proposed methods can be extended to every current signal in the HV system of an EV. Furthermore, we inverse the sign of i_{HVB} from now on to make its shape similar to the one of the EM. Thus, we can treat the HVB current signal as a delayed or preceded version of the EM, respectively.

Our goal is to find the time delay t_d in a bus system which can be described as

$$\begin{aligned} x_1(t) &= i_1(t) + n_1(t) \\ x_2(t) &= i_2(t - t_d) + n_2(t - t_d), \end{aligned} \tag{1}$$

where t stands for the time step, $x_1(t)$ is the measurement signal of the faster component, $x_2(t)$ describes the slower component's signal, $i_1(t)$ and $i_2(t)$ describe the corresponding currents and $n_1(t)$ and $n_2(t)$ are noise terms [15]. As we cannot retrieve the currents $i_1(t)$ and $i_2(t)$ directly, we cannot minimize the difference between $i_1(t)$ and $i_2(t)$. Instead, we directly minimize the difference between the two measurement signals $x_1(t)$ and $x_2(t)$.

3.1. Adaptive Filter

The idea of Emadzadeh et al. is to model the time delay as Finite Impulse Response (FIR) filter. They define $x_1(t)$ to be the faster signal. For each measurement $x_2(t_i)$ at time step t_i, they collect a row of the last M measurements of the other signal

$$\boldsymbol{x}_1(t_i - M + 1 : t_i) = [x_1(t_i - M + 1), x_1(t_i - M + 2), \dots x_1(t_i - 1), x_1(t_i)]\,. \tag{2}$$

Then, the authors search for an optimal channel impulse response vector $\boldsymbol{\omega}^*$ such that the deviation between $x_2(t_i)$ and $\boldsymbol{x}_1(t_i - M + 1 : t_i)\boldsymbol{\omega}^*$ becomes minimal. Mathematically, this can be expressed by the minimization of the expectation value of the Mean Squared Error (MSE) between the measurement value of the slower signal and the filtered measurement row of the faster signal. It results in the formula

$$\boldsymbol{\omega}^* = \underset{\boldsymbol{\omega}}{\operatorname{argmin}}\, E\left[\left(x_2(t_i) - \boldsymbol{x}_1(t_i - M + 1 : t_i)\boldsymbol{\omega}\right)^2\right]. \tag{3}$$

This is similar to our VM approach (see Section 3.3) with the difference that we minimize the variance instead of the MSE. In Emadzadeh's work, the optimal factor ω^* is estimated recursively. For the recursion, the authors implement and compare the four algorithms Least Mean-Squares (LMS), Normalized Least Mean-Squares (NLMS), Least Mean-Fourth (LMF) and Recursive Least-Squares (RLS). The optimal time delay estimate t_d^* is then the one where the impulse response $\boldsymbol{\omega}^*$ reaches its maximum, or mathematically

$$t_d^* = \underset{i\in[1,M]}{\operatorname{argmax}}\, \boldsymbol{\omega}^*(i) - 1. \tag{4}$$

For further details on this approach, we kindly refer the interested reader to [15].

3.2. Linear Regression

Our first approach is to use LR to identify the *basis functions* of two received signals and compare the horizontal offset between these functions. As degree of the basis function, we choose a parabola for two reasons. First, the sampling frequency of our measurements is high enough to fit the signals with a parabola for a short time duration. Second, the comparison of the horizontal offset is easiest with a parabola because it only has one extremum.

We collect the last M measurements x_k of the HV components $k \in \{\mathrm{EM}, \mathrm{HVB}\}$ in a measurement vector $\boldsymbol{y_k} = \begin{pmatrix} x_k(t) & x_k(t-1) & \cdots & x_k(t-M+1) \end{pmatrix}$. Then, we retrieve the weight vector $\boldsymbol{w_k} = \begin{pmatrix} w_{k,0} & w_{k,1} & w_{k,2} \end{pmatrix}^T$ with LR [22] according to

$$\boldsymbol{w_k} = \left(\sum_{n=1}^{N} \boldsymbol{\phi}^n (\boldsymbol{\phi}^n)^T \right)^{-1} \sum_{n=1}^{N} y_k^n \boldsymbol{\phi}^n. \tag{5}$$

Here, the notation y_k^n and $\boldsymbol{\phi}^n$ represents the n-th column of $\boldsymbol{y}_k$ and $\boldsymbol{\phi}$, respectively. The so-called *design matrix*

$$\boldsymbol{\phi} = \begin{pmatrix} 1 & t & t^2 \\ 1 & t-1 & (t-1)^2 \\ \vdots & \vdots & \vdots \\ 1 & t-M+1 & (t-M+1)^2 \end{pmatrix}^T$$

consists of $N = 3$ columns in our case.

With the weight vector from (5), we are able to fit a parabola

$$f_k(t) = w_{k,0} + w_{k,1} t + w_{k,2} t^2 \tag{6}$$

as basis function to the measurement vector $\boldsymbol{y_k}$.

After retrieving the basis functions in (6), we transfer them into vertex form

$$f_k(t) = w_{k,2}(t - x_{k,\mathrm{vertex}})^2 + y_{k,\mathrm{vertex}} \tag{7}$$

to identify the coordinates $(x_{k,\mathrm{vertex}}, y_{k,\mathrm{vertex}})$ of each basis function's vertex. The estimated time delay between the EM and the HVB current signals is then given by the difference on the time axis between their vertices according to

$$t_d^* = x_{\mathrm{EM,\,vertex}} - x_{\mathrm{HVB,vertex}}. \tag{8}$$

3.3. Variance Minimization

Our second approach is to minimize the variance of the difference between two signals $x_1(t)$ and $x_2(t)$ by shifting the signal $x_2(t)$ forward.

Like in Section 3.2, we collect the two signals $x_1(t)$ and $x_2(t)$ for M time steps. A straightforward idea for the minimization of the difference between $x_1(t)$ and $x_2(t)$ is to minimize their estimated MSE

$$\mathrm{MSE}(x_1(t), x_2(t)) = \frac{1}{M - T_{\max}} \sum_{t=1}^{M-T_{\max}} (x_1(t) - x_2(t + t_{d,i}))^2 \tag{9}$$

with different time shifts $t_{d,i}$ in a pre-defined range $t_{d,i} \in [T_{\min}, T_{\max}]$ with $T_{\max} < M$. However, our experiments show that we need a relatively high M to achieve stable results. We can significantly minimize M, if we take the estimated expected value

$$E = \frac{1}{M - T_{\max}} \sum_{t=1}^{M-T_{\max}} (x_1(t) - x_2(t + t_{d,i})) \,. \tag{10}$$

into account. Thus, instead of minimizing the MSE from (9), we minimize the estimated variance of the difference between the two signals

$$\sigma^2 = \frac{1}{M - T_{\max}} \sum_{t=1}^{M-T_{\max}} ((x_1(t) - x_2(t + t_{d,i})) - E)^2. \tag{11}$$

The time delay between the EM and the HVB current signals is the $t_{d,i}$ that minimizes the variance

$$t_d^* = \underset{t_{d,i}}{\operatorname{argmin}}\, \sigma^2. \tag{12}$$

As we do not know in the beginning whether $x_{\mathrm{EM}}(t)$ or $x_{\mathrm{HVB}}(t)$ is the faster signal, we have to choose one of them as $x_1(t)$ and the other one as $x_2(t)$ for the first execution and try $T_{\min} = -T_{\max}$. From the second execution on, the value of $T_{\min}$ and $T_{\max}$ can be reduced and chosen recursively, because the EV's bus system usually changes its time delay only once in the beginning, but not during advanced execution. Therefore, we choose $T_{\min}(t) = t_d(t-1) - 1$ and $T_{\max}(t) = t_d(t-1) + 1$ from the algorithm's second execution on.

3.4. Optimized Variance Minimization

As the results of our experiments (see Section 5) show, the VM concept provides the best results in terms of RMSE, run-time and required frame size. However, when running this concept in real time (both on simulated and real data, see Section 5.4), we find that the TDE is unstable and that the estimated time delay frequently alternates between different values. These many changes of the estimated time delay contradict the fact that the time delay is rather stable in reality, and that, if any, changes occur after relatively long periods. Thus, in order to stabilize the TDE, we suggest the following improvement of the *plain* VM approach from Section 3.3.

The main idea of the stabilization is to use a statistical test. The test's purpose is to quantify the *reliability* of the input data segment on which the TDE is performed. In fact, we know from Section 3.3 that the estimated time delay t_d at time step t minimizes the variance given by Equation (11). This equation in turn is based on the M last values of both signals. Due to the noise in the data, the estimation for the next step can jump to a different value, even if the vast majority of data points $(M-1)$ are shared between the two steps. The idea is thus to compare at each step the minimal variance with the second smallest one. If the difference between both in relative terms is not sufficiently large, we presume that the TDE is not reliable, and consequentially do not estimate a time delay. In this case, we simply keep the prediction from the last step. Otherwise, we update the estimation to the newly calculated t_d.

Formally, at each time step, we calculate the variance criterion from Equation (11) for each potential time delay $t_{d,i}$. Let us denote this by $\sigma^2(t_{d,i})$. Then, we know that the least achievable variance is given by

$$\sigma^2(t_d^*) = \min_{t_{d,i}} \sigma^2(t_{d,i}). \tag{13}$$

The second smallest achievable variance in turn is given by

$$\sigma^2(t_d^{**}) = \min_{t_{d,i} \neq t_d^*} \sigma^2(t_{d,i}). \tag{14}$$

In other words, we minimize the variance over all potential time delays except that which minimizes it (t_d^*). By definition, we have $\sigma^2(t_d^{**}) \geq \sigma^2(t_d^*)$. The intuition is that if the difference between those two values is not large enough, the noise makes it impossible to tell with high confidence which one is the real minimizer. The minimum in Equation (13) might result in t_d^* by random noise instead of being the true minimum. Thus, we suggest deciding whether to perform an update based on the criterion

$$\frac{\sigma^2(t_d^{**}) - \sigma^2(t_d^*)}{\sigma^2(t_d^*)} > K, \tag{15}$$

where K is a hyper-parameter defining the minimal percentage error required to perform an update. Clearly, the larger K, the more severe is the criterion, and the fewer updates are done. Therefore, we choose K to strike a balance between reliability on the one hand, and being up-to-date on the other hand. In fact, if we choose K too high, updates are performed only rarely, so that we can miss changes in the underlying real time delay. If K is chosen too small, then the predictions are more unstable. We empirically found $K = 0.2$ to strike a balance between both criteria for our power train data.

4. Experimental Setup

In this section, we explain the data and the setup for the experiments to evaluate the performance of the three concepts for TDE and the optimization presented above.

4.1. Data

For the evaluation of the three concepts and the optimization shown in Section 3, we use 74 data sets. The data sets contain all currents of the HV system and are recorded during representative drives on public roads with close to production EVs. We use bus loggers to record the data. The loggers store the received measurement signals from all ECUs and write them to a log file during each time step. After driving, we use the log files to execute our experiments and evaluate our approaches. Thus, the algorithms get at each time step the same input data which they would receive during execution on an ECU in the real EV. The recordings correspond to 10 h 33 min of driving. For the experiments, the 74 data sets are divided into 409 sub-data sets with a maximum length of 10,000 time steps. The minimum length among the 409 sub-data sets is 1,807 time steps.

For the Optimized VM approach, we create an additional data set artificially. The artificial data set bases upon the real data sets described above. However, instead of computing the time delay between two real signals, we introduce an artificial signal. This artificial signal is a real signal shifted by some time steps. We can then compute the time delay between the original signal and its artificially delayed correspondence. This has the advantage that we exactly know the time delay and thus know the ground truth.

4.2. Experiments

According to *Kirchhoff's current law*, we assume that the sum of the measurements of i_{HVB}, i_{EM}, i_{heat}, i_{cool} and i_{DCDC} is zero. Thus, we estimate the *ground truth* of the time delay for each real data set by minimizing the MSE of the complete data set (see Equation (9)). In this case, M is the length of the

data set, T_{max} is 10 time steps and T_{min} is -10 time steps, since the time delay in the EV is normally smaller than ten time steps.

In the experiment of the AF concept, we choose the frame size of 2^{10} as proposed by [15]. Additionally, we execute our experiments with a more efficient frame size of 2^8. We assume the length of the AF to be 10. All the other parameters for the used algorithms are chosen equally to [15]. Furthermore, we evaluate five different learning rates for LMS.

For the evaluation of the VM concept we test different frame sizes $M \in \{30, 50, 100, 200, 300\}$. In the first calculation, we also choose $T_{max} = 10$ and $T_{min} = -10$. From the second execution on we select $T_{min}(t) = t_d(t-1) - 1$ and $T_{max}(t) = t_d(t-1) + 1$.

For the Optimized VM, we choose a fixed frame size of 50 time steps. This frame size proved to be the best compromise between run-time and accuracy in previous experiments as described in subsection 5.3.

For each of the three concepts, we calculate the time delay every 20th time step. In total, this results in around 90,000 time delay estimations for each concept.

4.3. Environment

All concepts are implemented in Matlab R2015b with Microsoft Windows 10 on an HP® EliteBook™840 G3 with an Intel® Core™i5-6300U 2.40GHz CPU and 8 GB RAM.

5. Results

In this section, we present the results of our experiments and evaluate the performance of the three concepts and the optimization individually. The results of all three algorithms compared next to each other can be found in the next section.

5.1. Adaptive Filter

Based on the learning rate and parameters in [15], the RLS algorithm performs better than the LMS, NLMS and LMF algorithms (see Table 1).

Table 1. RMSE and run-time analysis of the AF concept for different algorithms with a frame size of 2^{10}.

	RMSE	Average Run-Time (s)
LMS	2.8972	$1.13 \cdot 10^{-2}$
NLMS	2.5163	$1.31 \cdot 10^{-2}$
LMF	2.6138	$1.14 \cdot 10^{-2}$
RLS	2.276	$1.07 \cdot 10^{-2}$

Furthermore, we analyze the learning rate for the LMS algorithm. As mentioned in [15], the learning rate μ is typically chosen in the range $0 < \mu < 2/(M\sigma_u^2)$, where σ_u^2 is the input signal variance and M is the length of the filter. Thus, we compare the performance of LMS with different $\mu = a/(M\sigma_u^2)$ and $a \in \{0.01, 0.05, 0.1, 0.5, 1\}$. In Table 2, the RMSE has the minimal value of 2.1479 with $a = 0.1$. It is much smaller than the RMSE of 2.8972 with the fixed learning rate in [15]. For a too large or a too small a, the performance of the LMS decreases significantly. This result is expected, since a too small learning rate leads to slow convergence while a large one most often misses the optimum.

Table 2. RMSE and run-time analysis of LMS with different learning rates and a frame size of 2^{10}.

	RMSE	Average Run-Time (s)
$a = 0.01$	3.0451	$1.14 \cdot 10^{-2}$
$a = 0.05$	2.3608	$1.14 \cdot 10^{-2}$
$a = 0.1$	2.1479	$1.14 \cdot 10^{-2}$
$a = 0.5$	2.6484	$1.14 \cdot 10^{-2}$
$a = 1$	3.2474	$1.14 \cdot 10^{-2}$

In addition, we evaluate the LMS algorithm with a more efficient frame size of 2^8. As printed in Table 3, the smaller frame size improves the run-time. Although some non-optimal learning rates improve their estimation accuracy, which we explain with the drop of local minima due to the shortened frame, the two best learning rates in the experiment with the frame size of 2^{10} increase their estimation errors with the smaller frame size.

Table 3. RMSE and run-time analysis of the LMS algorithm with a shorter frame size of 2^8.

	RMSE	Average Run-Time (s)
$a = 0.01$	2.8603	$2.80 \cdot 10^{-3}$
$a = 0.05$	2.5598	$2.80 \cdot 10^{-3}$
$a = 0.1$	2.3994	$2.80 \cdot 10^{-3}$
$a = 0.5$	2.4983	$2.80 \cdot 10^{-3}$
$a = 1$	3.0781	$2.80 \cdot 10^{-3}$

5.2. Linear Regression

Our first approach LR is, to a large extent, affected by noise and the offset between the two signals caused by measurement inaccuracies. Especially this offset leads to an imprecise estimation of the vertices and thus a wrong estimated time delay. Figure 5 shows an example for such a wrong estimation. In this data set, the time delay between i_{HVB} and i_{EM} is equal to 6 time steps. We train both curves on 200 measurement samples of their corresponding signal. However, due to noise and some vertical offset between the signals the vertex of the slower signal is not only shifted to the right but also to the top. The shift in vertical direction also affects the horizontal position of the vertex and results in a wrong TDE of 43 time steps.

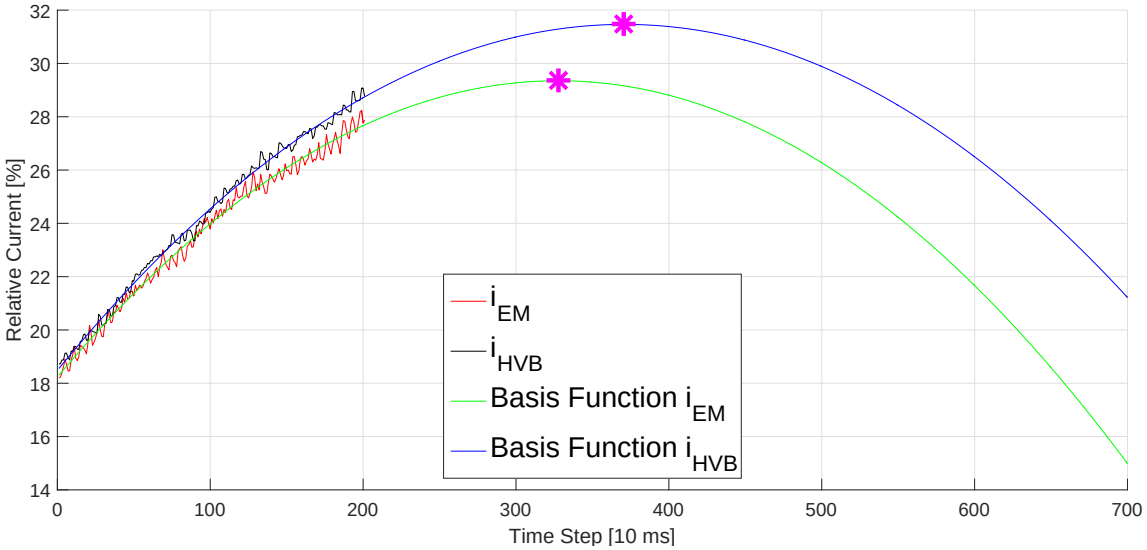


Figure 5. Basis functions of i_{HVB} (black) and i_{EM} (red) simulated by LR (blue and green, respectively). The magenta marked points are the vertexes. Their horizontal difference is 43 time steps in contrast to the real time delay which is 6 time steps. The wrong TDE is caused by the noise and the vertical offset of the measurement signals.

Table 4 shows the results and the average run-time of this approach with three different frame sizes. The run-time grows with increasing frame sizes, whereas the RMSE becomes smaller. Nevertheless, the RMSEs are in general very high even for large frames.

Table 4. RMSE and run-time analysis of the LR concept.

	RMSE	Average Run-Time (s)
Frame Size 30	$5.83 \cdot 10^{10}$	$1.74 \cdot 10^{-4}$
Frame Size 200	$4.92 \cdot 10^{4}$	$9.80 \cdot 10^{-4}$
Frame Size 300	$4.47 \cdot 10^{4}$	$1.40 \cdot 10^{-3}$

5.3. Variance Minimization

Table 5 shows the RMSE between the ground truth of the time delay and the calculated time delay. Furthermore, the table shows the average of the run-time for each time delay calculation, corresponding to different frame sizes M (in Equation (11)). We see that the concept requires a relatively short run-time as it benefits from the recursive calculation only in the area $t_{d,i} \in [T_{\min}, T_{\max}]$ with $T_{\min}(t) = t_d(t-1) - 1$ and $T_{\max}(t) = t_d(t-1) + 1$. In addition, the RMSE decreases while the size of the frame increases. The accuracy has a large improvement when the frame is enlarged from 30 time steps to 50 time steps.

Table 5. RMSE and run-time analysis of the VM concept.

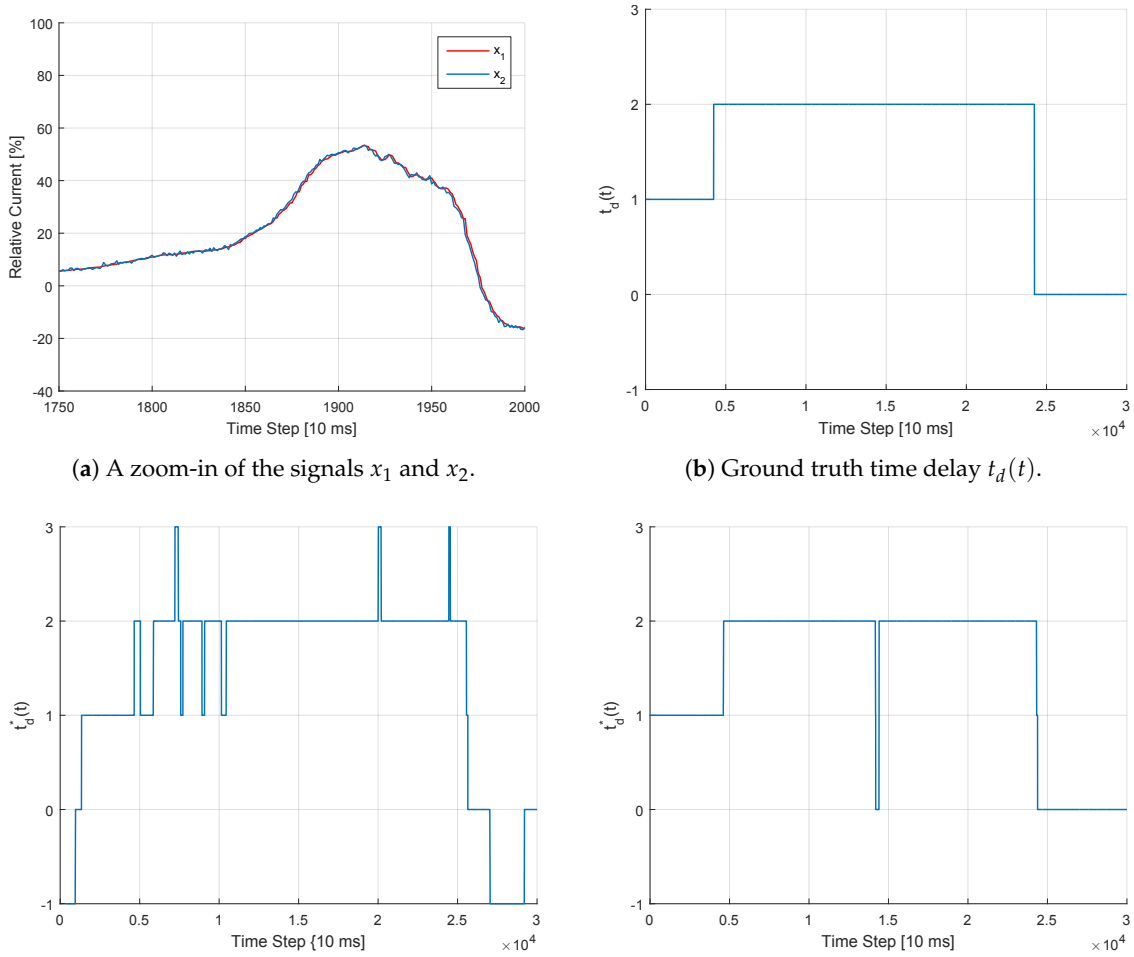
	RMSE	Average Run-Time (s)
Frame Size 30	2.0696	$4.54 \cdot 10^{-5}$
Frame Size 50	1.3034	$4.70 \cdot 10^{-5}$
Frame Size 100	1.2364	$4.77 \cdot 10^{-5}$
Frame Size 200	1.2215	$5.16 \cdot 10^{-5}$
Frame Size 300	1.1825	$5.79 \cdot 10^{-5}$

5.4. Optimized Variance Minimization

We evaluate the proposed stabilization approach twofold. First, we evaluate it based on simulated data with known ground truth time delay. Second, we evaluate the approach on real signals. While the first experiment shows the accuracy of the proposed approach, the second one shows its effectiveness in providing more stability.

5.4.1. Evaluation with Simulated Signals

In this experiment, we first take a current signal $x_1(t)$ from a real-world data set recorded on-board of an EV. Based on x_1, we then create a second signal $x_2(t) = x_1(t - t_d(t)) + n_2(t)$. Therefore, the ground-truth time delay $t_d(t) \geq 0$ is a realization of a random jump process that in known in advance. Furthermore, $n_2(t)$ is a white noise process whose variance is chosen such that the resulting Signal-to-Noise Ratio (SNR) is equal to -10. We then run our VM algorithm with and without stabilization to detect the delay $t_d(t)$. Figure 6 shows the results of this experiment.



(**a**) A zoom-in of the signals x_1 and x_2.

(**b**) Ground truth time delay $t_d(t)$.

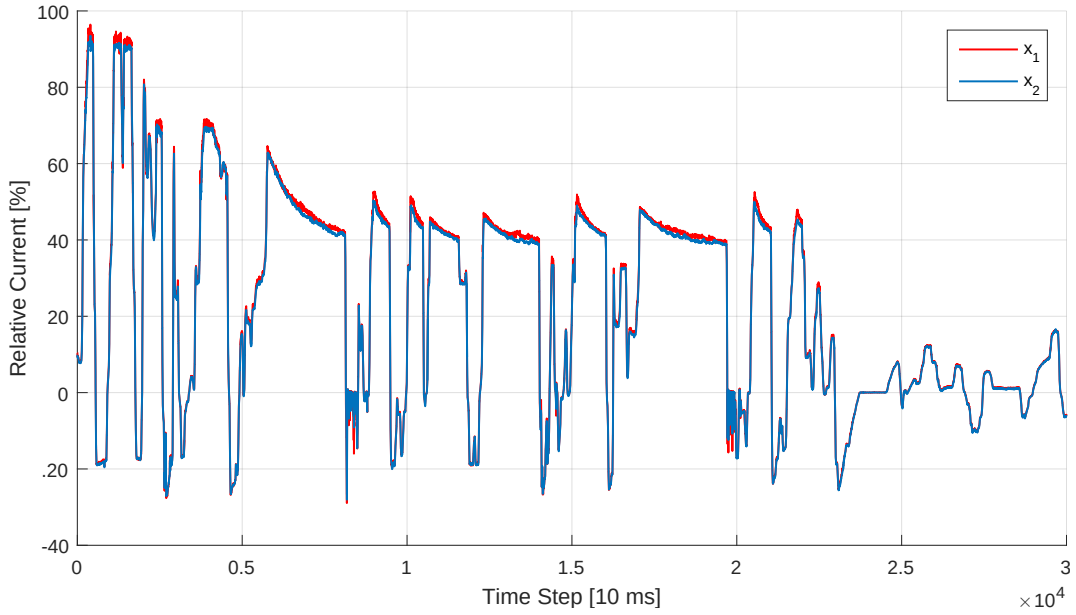
(**c**) Estimated time delay without stabilization.

(**d**) Estimated time delay with stabilization.

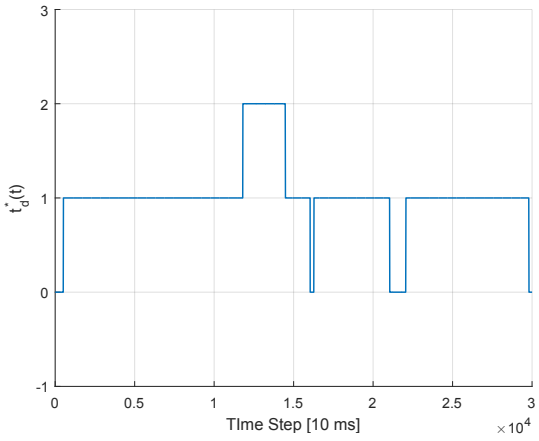
Figure 6. Illustration of the TDE estimation procedure using the VM approach with and without stabilization for the case of simulated signals. Clearly, the estimation is more stable when using the criterion in equation (15). Waiting for the *right* moment to perform an update comes however with the expense of a slightly delayed, yet more reliable, prediction. For example, the jump of $t_d(t)$ from 1 to 2 was detected with a delay of around 320 steps, which corresponds to around 3.2 seconds.

5.4.2. Evaluation with Real Signals

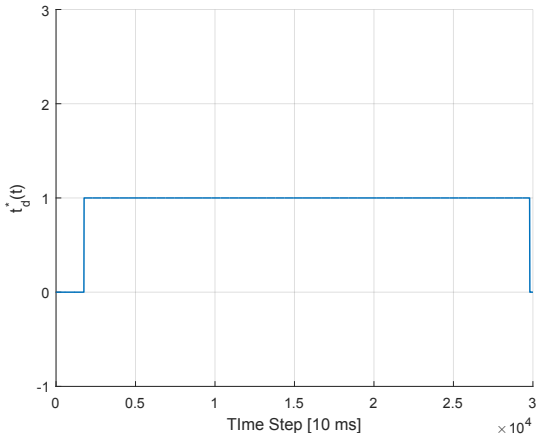
In this experiment, we take both signals $x_1(t)$ and $x_2(t)$ from a real-world data set. We then run the VM approach with and without stabilization, and plot the results in Figure 7.



(**a**) The signals x_1 and x_2.



(**b**) Estimated time delay without stabilization.



(**c**) Estimated time delay with stabilization.

Figure 7. Illustration of the TDE estimation procedure using the VM approach with and without stabilization for the case of real signals. Although we do not know the underlying true time delay, it is again clear that the TDE is more stable using the suggested approach.

6. Discussion

We discuss the advantages and drawbacks of the previously described and evaluated concepts in this section.

Although it is not as efficient and accurate as VM, the AF approach still retrieves better results than LR. The best results for AFs, in our case, are reached with the LMS algorithm and a learning rate of $\mu = 0.1/(M\sigma_u^2)$ (see Table 2). The learning rate, which must be chosen manually, is one drawback of this algorithm. It can lead to sub-optimal learning if the user chooses a wrong value. In contrast with LMS, the RLS algorithm does not require a learning rate. However, we see that the RLS algorithm has lower accuracy, requires longer run-time and more memory for a larger frame than the VM concept (see Table 6).

Table 6. RMSE, run-time analysis and frame size of all three concepts compared to each other.

	RMSE	Average Run-Time (s)	Frame Size
Adaptive Filter	2.3994	2.80×10^{-3}	2^8
Linear Regression	4.92×10^4	9.80×10^{-4}	200
Variance Minimization	1.3034	4.70×10^{-5}	50

LR has the advantage that it can directly find out the faster component. Thus, one single execution during the same time step for the same signal is sufficient even in the beginning, which makes it interesting, if a computational effective approach is needed. However, its efficiency suffers from the matrix inversion in Equation (5). Even worse, it is the least accurate of the three proposed concepts due to noise and vertical offsets between the signals. The high estimation errors make this approach unfeasible for our purpose. Another drawback is that a parabola is not always the optimal basis function for the regression of measurement signals.

The VM approach does not require such a basis function. Unfortunately, it is not able to detect the faster signal without trying each possible time delay for both signals. This results in a computationally expensive brute force calculation in the first time step. Afterwards, it is very efficient because it must only execute basic math operations and searches only for a restricted number of possible delays. Compared to the other approaches, VM requires the smallest frame size to retrieve feasible results. In total, Table 6 shows clearly that VM is the most accurate and fastest of the three proposed approaches with the lowest memory consumption.

For the high precision and low run-time, we decide to continue our work with the VM concept. Before we are able to apply our approach to series production EVs, we require further optimization as shown in Section 3.4 to stabilize the estimated time delay. This stabilization comes with another drawback. The algorithm requires more time steps to pass before it adapts to a new delay. Nevertheless, regarding that succeeding power train control functions require stable inputs, this drawback seems acceptable for us. Another drawback of the optimization is the threshold value K in Equation (15) which must be chosen manually. Although it does not require expert knowledge but can be set by trial and error, we would prefer an automated way for finding the optimal value for K.

7. Conclusions and Outlook

This article presents three different approaches for TDE of measurement signals in the power train of EVs. As automotive ECUs are designed very efficiently, our evaluation's focus lies also on computation and memory complexity and not solely on accuracy. Unfortunately, LR is not suited for our purposes because it suffers too much from vertical offsets in the measurement data. However, with VM, we present a feasible approach for TDE of distributed sensor systems of EVs. AFs are also not suited because they require too large frame sizes and have lower accuracy than VM. We recommend using VM due to its high estimation accuracy and computational efficiency. As the output of VM is not stable enough to directly process it to power train control functions of series EVs, we optimize it first. For the optimization, we introduce a threshold value as additional requirement for changing the value of the estimated time delay. The new requirement decelerates the detection of changed time delays. Nevertheless it improves the TDE's stability and accuracy.

After the introduction of an automated TDE system, we now know each signal's delay. However, if we correct the delay, we move some signals to the past and lose the measurements corresponding to the latest time steps. This is correct because in fact we do not receive up-to-date measurements, only delayed ones from the past. We really do miss the last measurements and there is a gap between the last received measurement and the present time step. Thus, our next work focuses on possible ways to close this gap by replacing the hidden information about the missing measurements from the latest time steps.

8. Patents

The TDE for the power trains of EVs is registered at the German Patent and Trade Mark Office (DPMA) as patent application. Both the VM approach as well as its RMSE-based version for TDE in the power trains of EVs are registered there as a common patent application. The optimization is registered as a third patent application resulting from this work.

Author Contributions: conceptualization, J.P.; methodology, J.P., X.W. and A.A.; software, J.P. and X.W.; validation, X.W.; formal analysis, J.P. and X.W. investigation, J.P. and X.W.; resources, J.P. and X.W.; data curation, J.P.; writing–original draft preparation, J.P., X.W. and A.A.; writing–review and editing, J.P. and X.W.; visualization, J.P., X.W. and A.A.; supervision, J.P.; project administration, J.P.; funding acquisition, J.P. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Conflicts of Interest: The authors declare that the publication of data must be approved by the BMW Group. Besides that, there are no conflicts of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript, or in the decision to publish the results.

Abbreviations

The following abbreviations are used in this manuscript:

AF	Adaptive Filter
AR	Auto-Regressive
ARIMA	Auto-Regressive Integrated Moving Average
ECU	Electronic Control Unit
EM	Electric Machine
EV	Electric Vehicle
FCHEV	Fuel Cell Hybrid Electric Vehicle
FIR	Finite Impulse Response
GNSS	Global Navigation Satellite Systems
HV	High Voltage
HVB	High Voltage Battery
I	Integrated
IDAACS	IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications
LSTM	Long Short-Term Memory Neural Network
MA	Moving Average
MOS-ELM	Online Sequential Extreme Learning Machine with Memory principle
LMF	Least Mean-Fourth
LMS	Least Mean-Squares
LR	Linear Regression
MSE	Mean Squared Error
NLMS	Normalized Least Mean-Squares
N4SID	Numerical State Space Subspace System Identification
RLMS	Recursive Least Mean-Squares
RLS	Recursive Least-Squares
RMSE	Root Mean Square Error
SNR	Signal-to-Noise Ratio
PCA	Principle Component Analysis
TDE	Time Delay Estimation
VM	Variance Minimization

References

1. Dixon, J.; Anderson, P.B.; Bell, K.; Træholt, C. On the ease of being green: An investigation of the inconvenience of electric vehicle charging. *Appl. Energy* **2020**, *258*. doi:10.1016/j.apenergy.2019.114090. [CrossRef]
2. Komarnicki, P.; Haubrock, J.; Styczynski, Z.A. Electrical components of an EV (Elektrische Komponenten des E-Kfz). In *Elektromobilität und Sektorenkopplung*; Springer: Berlin/Heidelberg, Germany, 2018; pp. 61–109.
3. Pfeiffer, J.; Wolf, P.; Pereira, R. A Fleet-Based Machine Learning Approach for Automatic Detection of Deviations between Measurements and Reality. In Proceedings of the 2019 IEEE Intelligent Vehicles Symposium (IV), Paris, France, 9–12 June 2019; pp. 2086–2092.
4. Guo, Y.; Zhang, L.; Zhao, J.; Wen, F.; Salam, A.; Mao, J.; Li, L. Networked Control of Electric Vehicles for Power System Frequency Regulation with Random Communication Time Delay. *Energies* **2017**, *10*, 621. doi:10.3390/en10050621. [CrossRef]

5. Kali, Y.; Ayala, M.; Rodas, J.; Saad, M.; Doval-Gandoy, J.; Gregor, R.; Benjelloun, K. Current Control of a Six-Phase Induction Machine Drive Based on Discrete-Time Sliding Mode with Time Delay Estimation. *Energies* **2019**, *12*, 170. doi:10.3390/en12010170. [CrossRef]
6. Zeng, H.; Di Natale, M.; Giusto, P.; Sangiovanni-Vincentelli, A. Statistical analysis of Controller Area Network message response times. In Proceedings of the 2009 IEEE International Symposium on Industrial Embedded Systems, Lausanne, Switzerland, 8–10 July 2009; pp. 1–10.
7. Lourtie, I.M.G.; Moura, J.M.F. Optimal Estimation of Time-Varying Delay. In Proceedings of the 1988 International Conference on Acoustics, Speech, and Signal Processing (ICASSP), New York, NY, USA, 11–14 April 1988.
8. He, H.; Chen, J.; Benesty, J.; Zhang, W.; Yang, T. A class of multichannel sparse linear prediction algorithms for time delay estimation of speech sources. *Signal Process.* **2020**, *169*, 107395. doi:10.1016/j.sigpro.2019.107395. [CrossRef]
9. Svilainis, L.; Aleksandrovas, A.; Lukoseviciute, K.; Eidukynas, V. Investigation of the Time of Flight estimation errors induced by neighboring signals. In Proceedings of the 2013 IEEE 7th International Conference on Intelligent Data Acquisition and Advanced Computing Systems (IDAACS), Berlin, Germany, 12–14 September 2013; pp. 413–418.
10. Mirzaei, M.; Asif, A.; Rivaz, H. Combining Total Variation Regularization with Window-Based Time Delay Estimation in Ultrasound Elastography. *IEEE Trans. Med. Imaging* **2019**, *38*, 2744–2754. doi:10.1109/TMI.2019.2913194. [CrossRef] [PubMed]
11. Hashemi, H.S.; Rivaz, H. Global Time-Delay Estimation in Ultrasound Elastography. *IEEE Trans. Ultrason. Ferroelectr. Freq. Control* **2017**, *64*, 1625–1636. doi:10.1109/TUFFC.2017.2717933. [CrossRef] [PubMed]
12. Garcez, C.C.R.; de Lima, D.V.; Miranda, R.K.; Mendonça, F.; da Costa, J.P.C.L.; de Almeida, A.L.F.; de Sousa, R.T., Jr. Tensor-Based Subspace Tracking for Time-Delay Estimation in GNSS Multi-Antenna Receivers. *Sensors* **2019**, *19*, 5076. doi:10.3390/s19235076. [CrossRef]
13. Xie, T.; Jiang, H.; Zhao, X.; Zhang, C. A Wi-Fi-Based Wireless Indoor Position Sensing System with Multipath Interference Mitigation. *Sensors* **2019**, *19*, 3983. doi:10.3390/s19183983. [CrossRef] [PubMed]
14. Schmidhammer, M.; Gentner, C.; Siebler, B.; Sand, S. Localization and Tracking of Discrete Mobile Scatterers in Vehicular Environments Using Delay Estimates. *Sensors* **2019**, *19*, 4802. doi:10.3390/s19214802. [CrossRef] [PubMed]
15. Emadzadeh, A.A.; Lopes, C.G.; Speyer, J.L. Online time delay estimation of pulsar signals for relative navigation using adaptive filters. In Proceedings of the 2008 IEEE/ION Position, Location and Navigation Symposium, Monterey, CA, USA, 5–8 May 2008; pp. 714–719.
16. Liu, X.; Yang, X.; Xiong, W. A robust global approach for LPV FIR model identification with time-varying time delays. *J. Franklin Inst.* **2018**, *355*, 7401–7416. doi:10.1016/j.jfranklin.2018.07.025. [CrossRef]
17. Park, P.; Di Marco, P.; Shin, H.; Bang, J. Fault Detection and Diagnosis Using Combined Autoencoder and Long Short-Term Memory Network. *Sensors* **2019**, *19*, 4612. doi:10.3390/s19214612. [CrossRef] [PubMed]
18. Srinivasa Rao, D.; Siva Kumar, M.; Ramalinga Raju, M. Enhancement of robust performance for fuzzy parametric uncertain time-delay systems using differential evolution algorithm. *Int. J. Robust Nonlinear Control* **2019**, *29*, 6337–6356. doi:10.1002/rnc.4694. [CrossRef]
19. Shen, S.; Song, A.; Li, T.; Li, H. Time delay compensation for nonlinear bilateral teleoperation: A motion prediction approach. *Trans. Inst. Meas. Control* **2019**, *41*, 4488–4498. doi:10.1177/0142331219860928. [CrossRef]
20. You, F.; Cheng, S.; Zhang, X.; Chen, N. Robust fault estimation for Takagi-Sugeno fuzzy systems with state time-varying delay. *Int. J. Adapt Control Signal Process.* **2019**, doi:10.1002/acs.3073. [CrossRef]
21. Pfeiffer, J.; Wu, X. Automated Time Delay Estimation for Distributed Sensor Systems of Electric Vehicles. In Proceedings of the 2019 10th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS), Metz, France, 18–21 September 2019; pp. 609–614.
22. Barber, D. *Bayesian Reasoning and Machine Learning*; Cambridge University Press: Cambridge, UK, 2011.

3.2 Time Series Prediction

The Time Delay Estimation (TDE) proposed in the previous section allows to correct time delays between measurement signals from different ECUs according to Hypotheses H1 and H2a. But it does not allow to estimate not yet received measurement values from delayed signals. If I correct the time delay of t_d time steps of a delayed signal, I shift the signal t_d steps to the past. The shifting results in not available measurement values of the slower signal for the last t_d time steps. This corresponds to the real system behavior. Indeed, the *Ego ECU*, the ECU on which I am operating on, did not yet receive the unavailable measurement values. My idea is to estimate replacement values for the missing measurements with the help of time series prediction. In the following paper, Mohamed Ali Razouane and I compare different time series prediction algorithms and examine their suitability for estimating replacement values for delayed HV current measurements of EVs to validate Hypothesis H3. We evaluate and compare the following five different algorithms:

1. Simple Exponential Smoothing
2. Holt-Winters Exponential Smoothing
3. Auto-Regressive Integrated Moving Average (ARIMA)
4. Box-cox transformation, ARMA residuals, Trend and Seasonality (BATS)
5. Trigonometric seasonal, Box-cox transformation, ARMA residuals, Trend and Seasonality (TBATS)

To evaluate the accuracy of the algorithms above, we additionally implement two naive methods and compare the results:

1. Random Walk
2. Naive Method which simply takes the last value as prediction

Additionally, we use Bootstrap Aggregation to combine different algorithms.

Unfortunately, our results show that the advanced algorithms achieve only a relatively slightly improved accuracy with significantly higher computational effort than the naive methods. Thus, without additional research the prediction algorithms beyond the scope of this thesis, Hypothesis H3 cannot be validated for time series prediction on automotive ECUs. To predict HV currents on ECUs, I recommend to use the Naive Method. The Naive Method has the advantage with respect to Hypothesis H4 that it works without any parameters due to its simplicity. The more advanced algorithms require at least the number of training points to be set manually, and, depending on the individual algorithm, further additional parameters.

My contribution to the paper is the conceptualization. I formulate the problem statement and the introduction of the paper. Together with Mohamed Ali Razouane, I research into the state of the art and write the corresponding section in the paper. Mohamed Ali Razouane's contribution is the methodology. He researches, implements and evaluates appropriate time series prediction algorithms. Together, we decide which algorithms shall be published in the paper. Like in the previous papers, it is my responsibility to collect all the data used as input for the algorithms in the scope of this paper. Besides funding acquisition, administration and supervision of the project, I am responsible for reading and correcting all equations as well as all the text written in the paper.

In this context, I must mention that there are wrong indices in some equations in the paper even after my and the reviewers' inspection. Namely, in equations (7) and (12). In correct notation, the formulae are stated as

$$y_{t+1}^{(\omega)} = l_t + \phi b_t + \sum_{i=1}^{T} s_{t-m_i+1}^{(i)} + d_{t+1} \tag{3.1}$$

for equation (7) of the paper and

$$y_{t+1}^{(\omega)} = l_t + \phi b_t + \sum_{i=1}^{T} s_t^{(i)} + d_{t+1} \tag{3.2}$$

for equation (12) according to [4].

2020 Forum on Integrated and Sustainable
Transportation Systems (FISTS)
November 3-5, 2020, Delft - The Netherlands

Time Series Prediction for Measurements of Electric Power Trains

Jakob Pfeiffer[1,2] and Mohamed Ali Razouane[2]

***Abstract*— Real-time systems require up-to-date information. Measurement signals in the power train of Electric Vehicles (EVs) are however often received with individual time delays due to the distributed architecture of the power train. Our idea is to compensate the time delays by predicting each signal from the last received value until the present time step. In this work, we evaluate 5 state-of-the-art algorithms and 2 naive methods for time series prediction. We execute all algorithms on real power train data of EVs and compare the results. Our evaluation focuses on run-time and accuracy. All methods achieve a prediction error rate of less than 5 %. As expected, the benchmark naive method is the fastest. Surprisingly, it retrieves comparably accurate results as Exponential Smoothing. BATS and TBATS are the slowest methods. Nevertheless, they achieve the best accuracy, but suffer from outliers. Auto-Regressive Integrated Moving Average (ARIMA) achieves the smallest Mean Absolute Percentage Error (MAPE) and thus the best compromise between outliers and accuracy of all algorithms. Additionally, to further improve the accuracy, we investigate the benefits of combining predictions of different algorithms.**

I. INTRODUCTION

Modern vehicles consist of several distributed and embedded systems. For example, the climate control unit and the motor unit are separate embedded systems. Each of these systems consists of sensors and an Electronic Control Unit (ECU). The ECUs are connected and exchange information via bus systems. The bus communication requires time and thus, many measurements are delayed once they arrive at an ECU (see Fig. 1). Conversely, this means that from an ECU's perspective, measurements of the actual time step from other ECUs are not yet available (see Fig. 2).

The missing or delayed availability of measurements is problematic, because many vehicular real-time control functions depend on these data and their timeliness. Especially Electric Vehicles (EVs) lose efficiency and performance due to time delays [1]. While there exist already several strategies to detect the time delay in distributed sensor systems of EVs (e.g. [1], [2]), there is to the best knowledge of the authors no strategy to compensate not yet received measurement values.

Our goal in this work is to construct plausible virtual measurement values for not yet received signals. We take the already received measurements from past time steps and predict the values until the present.

Selected works in the field of time series prediction are presented in II. We analyze (see III) and evaluate (see V) 7 time series prediction algorithms. For the evaluation,

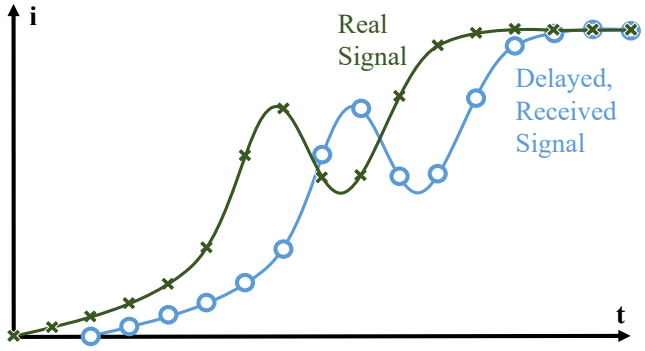


Fig. 1. Due to time delays between the distributed system in the power train of EVs, an ECU receives a delayed version (blue circles) of the original electric current measurement signal (green crosses).

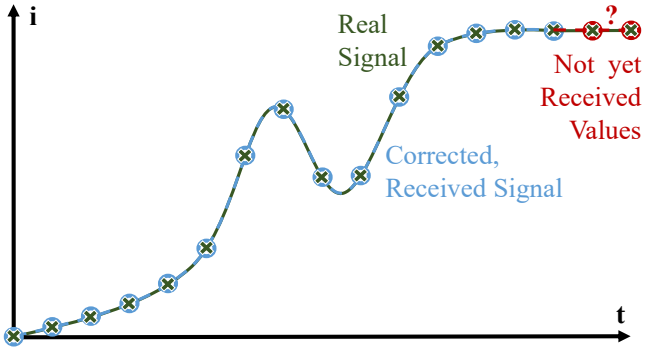


Fig. 2. If we correct the delayed signal according to [1], we realize that we did not receive the measurements of the last time steps (red circles) yet. Nevertheless, those actual measurements are needed for controlling the EV. Our goal in this work is to predict the missing values until the present.

we measure the performance of the algorithms in several experiments on real power train data of close to production EVs (see IV). In the end of this paper, we conclude the advantages and drawbacks of each algorithm (see VI).

II. STATE OF THE ART

Time series prediction is a common and often referred to problem [3]. Logically, there exists a plethora of literature about time series prediction, actually about time series prediction related to EVs. However, to the best of our knowledge, none of the existing literature deals with the prediction of High Voltage (HV) currents in the power trains of EVs. Our contribution is the analysis and evaluation of 5 state-of-the-art time series prediction algorithms for HV measurements in electric power trains. The focus of our evaluation lies not only on accuracy but also on run-time efficiency in order to enable an execution on automotive ECUs with limited memory and processing power. Previous

[1]Jakob Pfeiffer is with BMW Group, Petuelring 130, 80788 Munich, Germany Jakob.J.Pfeiffer@bmwgroup.com
[2]Both authors are with the Department of Electrical and Computer Engineering, Technical University of Munich, Arcisstr. 21, 80333 Munich, Germany {Jakob.Pfeiffer, Medali.Razouane}@tum.de

978-1-7281-9503-2/20/$31.00 ©2020 IEEE

measurements are the only input for our considered algorithms. The algorithms do not require additional information for the prediction. Thus, our time series prediction is not restricted to individual signals and can flexibly be applied to several measurement signals in the electric power train.

Styler et al. make use of time series prediction to develop an efficient discharging and charging strategy for the supercapacitor of an EV [4]. Their approach is based on data collected during real drives. However, in contrast to our data, Styler's data set only contains position data and no power train measurements. Thus, Styler's team has to simulate the HV currents which they use for their purposes.

The strategy for the energy management of Fuel Cell Hybrid Electric Vehicles (FCHEVs) proposed by Ibrahim et al. is based on wavelet transform and Auto-Regressive Integrated Moving Average (ARIMA) models [5]. Unfortunately, the authors do not mention how they access or generate the data. A difference to our approach is that our prediction horizon is shorter. We predict the next 20 time steps while Ibrahim et al. predict up to 64 time steps. Another difference is that we access the measurements of each HV component separately and predict the consumption component-wise. Ibrahim et al., in contrast, predict the total consumption of the EV and assign it to the components based on the frequencies of changes in the measurement signal.

Bolovinou et al. also predict the total consumption of the EV independent of the individual HV components [6]. They collect real drive data with an experimental EV. Based on the power train data and additional information about the EV's environment, they use the Baseline algorithm, Linear Regression and Support Vector Regression (SVR) to estimate the EV's future consumption and its remaining cruising range. Thus, their relatively large prediction horizon reaches until the next charging point. As stated above, our maximum prediction horizon of 20 time steps is much shorter.

Although Zulkas et al. state in the outlook of their paper that their approach can be extended to problems of the automotive domain, their focus is on the prediction of the energy consumption of their laboratory [7]. For the prediction, they implement and compare Auto-Regressive Moving Average (ARMA) models and Kalman Filters.

Independent from applications in the automotive context, there exist many attempts for feature-based model selection for univariate time series prediction. For instance, Collopy and Armstrong design a rule-based system [8]. For their system, they combine 4 different algorithms (Random Walk, Regression, and 2 types of Exponential Smoothing). The system predicts annual time series of demographic and economic data. Depending on 18 data features, the system decides which of the 4 prediction algorithms shall be chosen.

Shah uses a similar system with 3 algorithms [9]. The algorithms are 2 types of Exponential Smoothing and a basic structural time series model. Shah's system chooses the optimal algorithm depending on 26 data features. The author evaluates the approach on 4 data sets of the *M competition*. (For further information about the M competitions, see [3].)

Regarding the 2 previously mentioned publications on feature-based time series prediction, we find that although the researchers highlight the usefulness of time series features for selecting the best prediction algorithm, most of the existing approaches depend on the manual choice of an appropriate set of features. The manual choice complicates the algorithm selection and makes it error-prone, although there are works that categorize and automatically analyze features of time series [10]. Inspired by the work of Hatami et al. [11] and Wang and Oates [12], this paper aims to explore time series prediction based on model selection and model averaging.

In our previous work, we introduce and evaluate several approaches for time delay estimation [1], [2]. In this work, we assume that the time delay is already known as depicted in Fig. 2. Here, our goal is to close the gap between the last received measurement value and the present time step.

III. CONCEPTS

After researching into the state of the art in the previous section, we now take a closer look at the prediction algorithms considered for this work. Additionally, we outline Bootstrap Aggregation to improve the prediction results.

A. Prediction Algorithms

We consider the following 4 groups of algorithms.

1) Exponential Smoothing: Exponential Smoothing is a family of time series prediction algorithms first proposed more than 60 years ago [13]. In the scope of this work, we focus on the simple and the fully additive Holt-Winters model [14].

The basic idea of Exponential Smoothing is to construct predictions of future values $\hat{y}_{t+1}$ as weighted averages of past observations y_t and former predictions $\hat{y}_t$. The heavier weight is thereby assigned to the more recent values. Values from the more distant past are weighted less. Formally, the simple Exponential Smoothing prediction equation can be written as

$$\hat{y}_{t+1} = \alpha \cdot y_t + (1 - \alpha) \cdot \hat{y}_t, \tag{1}$$

where $0 < \alpha < 1$ is a smoothing factor.

An extension of this basic model is the fully additive Holt-Winters model. It predicts the value for the next time step by considering additive trend and seasonality. The seasonality aspects are included by extending the prediction from equation (1) by the 3 hidden state variables

$$\begin{aligned} l_t &= \alpha \cdot (y_t - s_{t-m}) + (1 - \alpha) \cdot (l_{t-1} + b_{t-1}), \\ b_t &= \beta \cdot (l_t - l_{t-1}) + (1 - \beta) \cdot b_{t-1}, \\ s_t &= \gamma \cdot (y_t - l_{t-1} - b_{t-1}) + (1 - \gamma) \cdot s_{t-m}, \end{aligned} \tag{2}$$

where l_t is the series level, b_t the trend, and s_t the seasonal component at time step t. α, β and γ are the corresponding smoothing coefficients. These are fitted by an optimization algorithm and have values between 0 and 1. m denotes the seasonality factor. It reflects the number of time steps in a seasonal period and ensures that the seasonality is correctly modeled. m can be obtained by means of a spectral density analysis of the simple and partial auto-correlation functions in conjunction. The new prediction is given by

$$\hat{y}_{t+1} = l_t + b_t + s_{t+1-m}. \tag{3}$$

2) ARIMA: ARIMA is a class of statistical models for analyzing and predicting time series data [15]. It is a generalization of the simpler ARMA extended by integration.

The acronym ARIMA is descriptive and captures the key aspects of the model itself. These can be resumed in the 3 following components:

1) Auto-Regressive (AR): A model that uses the dependent relationship between an observation and a number of lagged observations.
2) Integrated (I): Differentiation of raw observations to make the time series stationary. This can be achieved by subtracting an observation at the actual time step from an observation at the previous time step.
3) Moving Average (MA): A model making use of the dependency between an observation and residual errors from a moving average model applied to lagged observations.

Each of these components is explicitly specified in the model parameters in the standard notation ARIMA(p, d, q). They are substituted with integer values to indicate the specific model being used and defined as follows:

1) p: The number of lag observations included in the model, also called the lag order.
2) d: The number of times that the raw observations are differentiated, also called the degree of differentiation.
3) q: The size of the moving average window, also called the order of moving average.

The predicted value $\hat{y}_{t+1}$ of the future step $t+1$ is therefore a constant and a weighted sum of one or more recent values of y or of one or more recent values of the prediction error e. For our example, let $p = 1, d = 1, q = 2$. The ARIMA model obtained in this case is a damped-trend linear Exponential Smoothing. It extrapolates the local trend at the end of the time series. Simultaneously, it flattens the trend out at longer forecast horizons to introduce a note of conservatism. For the prediction, we first calculate the d^{th} difference $\hat{y}'_{t+1}$ of the future value y_{t+1}. The difference is a linear combination of past values of the original time series and past values of the prediction errors. It can be computed according to

$$\begin{aligned}\hat{y}'_{t+1} = l_t + \alpha_0 \hat{y}'_t + \alpha_1 \hat{y}'_{t-1} + ... + \alpha_p \hat{y}'_{t-p} \\ + e_{t+1} + \theta_0 e_t + ... + \theta_q e_{t-q},\end{aligned} \tag{4}$$

where l_t is the series level and e_t the prediction error at time step t. α_j is the slope coefficient relative to the d^{th} difference $\hat{y}'_{t-j}$ of y_{t-j} with $j \in \{1, 2, \dots p\}$. θ_k is the moving average parameter relative to the prediction error e_{t-k} with $k \in \{1, 2, \dots q\}$. e_{t+1} is hereby assumed as white noise. The integrated part of ARIMA is reflected in the d^{th} difference $\hat{y}'_{t-j}$ of $\hat{y}_{t-j}$. For a first differentiation, $\hat{y}'_t$ can for instance be obtained by

$$\hat{y}'_t = y_t - y_{t-1}, \tag{5}$$

where y_t and y_{t-1} are the true values, respectively at time step t and $t-1$. Finally, we retrieve the prediction equation

$$\hat{y}_{t+1} = y_t + \hat{y}'_{t+1}. \tag{6}$$

3) BATS and TBATS: Box-cox transformation, ARMA residuals, Trend and Seasonality (BATS) and Trigonometric seasonal, Box-cox transformation, ARMA residuals, Trend and Seasonality (TBATS) are an extension of the state-space modeling framework [16]. They introduce a comprehensive approach for predicting complex seasonal time series such as those with multiple seasonal periods, high frequency seasonality and non-integer seasonality. This is achieved by leveraging the benefits of Box-Cox transformations, Fourier representations with time varying coefficients, and ARMA error correction. The Box-Cox transformation solves the issues of non-linearity in the data. The ARMA model addresses the de-correlation of residuals in the time series data. De Livera et al. prove that BATS model can improve the prediction performance compared to simple state space models [16]. A key feature of both frameworks is that they rely on an optimized method that greatly reduces the computational complexity of the maximum likelihood estimation.

The BATS model is rooted in Exponential Smoothing (compare III-A.1). It reformulates equation (3) as

$$y^{(\omega)}_{t+1} = l_t + \phi \cdot b_t + \sum_{i=1}^{T} s^{(i)}_{t-m_i+1} + d_t \tag{7}$$

and the hidden state variables from (2) as

$$\begin{aligned} l_t &= l_{t-1} + \phi \cdot b_{t-1} + \alpha \cdot d_t, \\ b_t &= (1-\phi) \cdot b + \phi \cdot b_{t-1} + \beta \cdot d_t, \\ s^{(i)}_t &= s^{(i)}_{t-m_i} + \gamma_i \cdot d_t, \end{aligned} \tag{8}$$

with

$$d_t = \sum_{i=1}^{p} \varphi_i \cdot d_{t-i} + \sum_{i=1}^{q} \theta_i \varepsilon_{t-i} + \varepsilon_t. \tag{9}$$

Here, $y^{(\omega)}_t$ is the observation at time step t Box-Cox transformed with the parameter ω. Similar to III-A.1, $s^{(i)}_t$ denotes the i-th seasonal component, l_t the local level and b_t the dampened trend. The notation d_t stands for the ARMA(p, q) process for residuals. As we cannot compute the prediction error e_t directly, it is modeled as a Gaussian white noise process. e_{t-i} stands for the i-th Box-Cox transformed prediction error. The Box-Cox transformation parameter ω, the smoothing parameters α and β, the trend damping factor ϕ, the ARMA coefficients φ_i and θ_i, as well as the seasonal smoothing factor γ_i can all be estimated by the means of a Gaussian likelihood process.

TBATS extends the BATS model by including a trigonometric formulation for decomposing complex seasonal time series and identifying latent seasonal components [16]. The seasonal component is modeled based on a Fourier series as follows

$$\begin{aligned} s^{(i)}_t &= \sum_{j=1}^{k_i} s^{(i)}_{j,t}, \\ s^{(i)}_{j,t} &= s^{(i)}_{j,t-1} \cdot cos\lambda^{(i)}_j + s^{*(i)}_{j,t-1} \cdot sin\lambda^{(i)}_j + \gamma^{(i)}_1 \cdot d_t, \\ s^{*(i)}_{j,t} &= -s_{j,t-1} \cdot sin\lambda^{(i)}_j + s^{*(i)}_{j,t-1} \cdot cos\lambda^{(i)}_j + \gamma^{(i)}_2 \cdot d_t, \end{aligned} \tag{10}$$

where $\gamma_1^{(i)}$ and $\gamma_2^{(i)}$ are the smoothing parameters. $\lambda_j^{(i)}$ is retrieved by

$$\lambda_j^{(i)} = 2\pi j / m_i, \tag{11}$$

whereby m_i describes the length of the i-th seasonal period. $s_{j,t}^{(i)}$ denotes the stochastic level of the i-th seasonal component. $s_{j,t}^{*(i)}$ reflects the change in the seasonal component over time. The number of harmonics required for the i-th seasonal component is denoted by k_i. The trigonometric expression of seasonality terms helps to reduce the number of model parameters when the frequencies of seasonality are high. It also adds to the flexibility of the model in dealing with complex seasonality. The measurement equation for $y_{t+1}^{(\omega)}$ discussed in equation (7) is replaced by

$$y_{t+1}^{(\omega)} = l_{t-1} + \phi \cdot b_{t-1} + \sum_{i=1}^{T} s_{t-1}^{(i)} + d_t. \tag{12}$$

Point forecasts and forecast intervals for both the TBATS and the BATS models can be obtained using the inverse Box-Cox transformation of appropriate quantiles of the distribution of $y_{n+h|n}^{(\omega)}$. h is hereby the forecasting horizon and n the number of points in the observed data $\mathbf{y} = (y_1, ..., y_n)$.

4) Naive Methods: To benchmark the advanced prediction algorithms presented above and to show their effectiveness, we introduce 2 naive methods. The naive methods predict the next values of time series without further computational intelligence. The lack of computations or advanced models makes them very fast and proves if the predictions of the advanced methods are really accurate, or if the same results can be achieved with less effort.

The first naive method simply predicts the last measurement as future value

$$\hat{y}_{t+1} = y_t. \tag{13}$$

The second naive method is the driftless Naive Random Walk. A time series is said to follow a Random Walk if the differences from one observation to the next one are random. In other words, the series itself is not necessarily random but its first differences are. A Random Walk for a time series is written as

$$\hat{y}_{t+1} = y_t + \varepsilon_{t+1}, \tag{14}$$

where $\hat{y}_{t+1}$ is the predicted value at time step $t+1$, y_t its current value at time step t and ε_{t+1} is the unsystematic component and can be modeled as a white noise process [17].

B. Bootstrap Aggregation

Bootstrap Aggregation, commonly known as Bagging, is a method for reducing variance without increasing the bias of predictions. It enables to achieve more accurate predictions by combining forecasts of different predictors [18]. Combining predictions is especially useful when the uncertainty about the environment and the prediction method is relatively high, and when errors need to be moderated. In Bagging, predictors are trained on bootstrapped versions of the original data. The predictors form an ensemble. Predictions are generated by applying all predictors on the data set at hand and then combining the results. This can be achieved by averaging the obtained results, for example. Bagging tackles the three sources of uncertainties. It helps to moderate data uncertainty and the variation of the inherent random component that exist in time series. It also helps to temper the uncertainty linked with the selection of the optimal model form. Parameter uncertainty can furthermore be softened especially in terms of selecting the best set of parameters for describing the data. Bergmeir et al. show an example of successfully applying bagging for time series prediction [19].

IV. EXPERIMENTAL SETUP

In this section, we explain the circumstances of our experiments and how we evaluate the results.

A. Data Set

Our reference data consists of an extensive set of recordings of current measurements in power trains of EVs. The measurement data are recorded on public roads and reflect the behavior of power trains of EVs under common driving conditions. The data set covers in total $4\,\mathrm{h}$ of driving data measured at a frequency of $100\,\mathrm{Hz}$. For our experiments, we focus on the 5 HV currents of the HV battery, the electric machine, the electric heating, the air-conditioning compressor and the DCDC converter.

We divide the data set at hand into training and test segments to apply nested cross-validation. To this end, we split the data into chunks of constant size. Each chunk consists of 20 data points. In total, 10,480 chunks are taken into consideration.

To better estimate the prediction error of each algorithm, a common approach is to average the errors over all the train/test splits. The technique we use is based on a method called forward-chaining. It is referred to in the literature as rolling-origin evaluation [20] or rolling-origin-recalibration evaluation [21]. Based on this method, we successively consider each data chunk as the test set (see Fig. 3). All previous data is assigned to the training set. Assuming that the data set can be divided in 5 chunks as in Fig. 3, we produce 4 different training and test splits. By producing multiple different train/test splits, we achieve a better assessment of the prediction accuracy of each algorithm. The error on each split is again averaged in order to compute a robust estimate of each algorithm's error. The overall prediction error ϵ is modeled according to

$$\epsilon = \sum_{i=1}^{n} \sum_{j=1}^{m} \epsilon_j^{(i)}, \tag{15}$$

where n stands for the number of splits, m for the number of data points per split and $\epsilon_j^{(i)}$ for the error performance metric at hand.

B. Performance Metrics

To measure the performance of each algorithm in respect of its accuracy and computational efficiency according to (15), we introduce the following metrics.

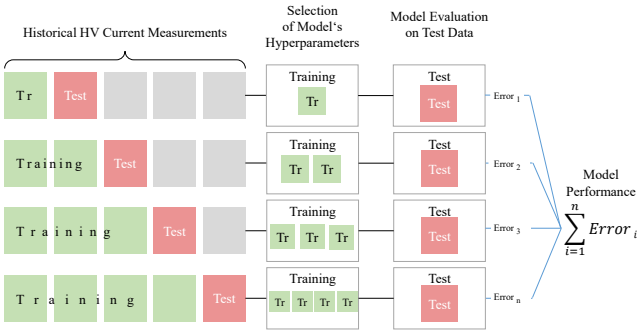


Fig. 3. Chronological representation of the nested cross-validation pipeline. The pipeline is used to split the data set into, in this exemplary case, 5 training and test chunks.

1) Root Mean Square Error: The Root Mean Square Error (RMSE) is a quadratic scoring rule that measures the average magnitude of the error. It is the square root of the average of squared differences between prediction $\hat{y_t}$ and actual observation y_t. It is given by

$$\mathrm{RSME}(y_t, \hat{y_t}) = \sqrt{\sum_{t=1}^{h} \frac{(y_t - \hat{y_t})^2}{h}}, \tag{16}$$

where h is the prediction horizon.

2) Mean Absolute Percentage Error: The Mean Absolute Percentage Error (MAPE) is a statistical measure of the accuracy of a prediction model. In a set of predictions divided by the actual values, the MAPE is the average error magnitude. The average error magnitude reflects the average over the test sample of the absolute differences between prediction and actual observation where all individual differences have equal weight. It is given by

$$\mathrm{MAPE}(y_t, \hat{y_t}) = \frac{100}{h} \sum_{t=1}^{h} \frac{|y_t - \hat{y_t}|}{|y_t|}. \tag{17}$$

3) Run-Time: Analyzing the run-time performance of each prediction algorithm is important for investigating its suitability for real-time systems. If the prediction takes too much time, it becomes obsolete. Therefore, we execute all algorithms several times on the same execution platform under the same circumstances and measure their run-time.

C. Environment

For the comparison within the scope of this paper, we implement and execute all algorithms in Python 3.6. with Microsoft Windows 10 as operating system on an HP® EliteBook™840 G3 with an Intel® Core™i5-6300U 2.40GHz CPU and 8 GB RAM.

D. Experiments

This work aims to find the best prediction algorithm among a pool of 7 candidate algorithms, or their best combination. Its essence is to link the knowledge on prediction errors of different algorithms to historical data. Therefore, this subsection focuses on the experiments applied for mapping historical current measurements to prediction method performances. In the first step, we focus on retrieving the optimal hyper parameters of each algorithm and assessing the individual performances (see IV-D.1). In the second step, we aim to assess the added value of combining forecasts from different methods in reducing uncertainty and increasing forecasting accuracy (see IV-D.2).

1) Individual Performance Assessment: The core focus in this section is to analyze the individual performance of the algorithms introduced in section III in predicting future HV current values. The analysis is performed under specific constraints. The main constraint is that the inference and prediction of future current values are based only on historical measurements. The performance of each algorithm is assessed using the nested cross-validation procedure described in IV-A.

In a first stage, we perform an exhaustive grid search through a manually specified subset of the hyper parameter space on the training splits. This helps to identify the best combination of hyper parameters for each algorithm. As a selection metric, we employ the Akaike Information Criterion (AIC). The AIC rewards the goodness of fit as assessed by the likelihood function given the number of estimated parameters k per algorithm. Simultaneously, k and the algorithm's complexity are penalized by the AIC to prevent over-fitting. Let $\hat{L}$ be the maximum value of the likelihood function for the prediction algorithm. The AIC value of the algorithm is then

$$\mathrm{AIC} = 2k - 2\ln(\hat{L}), \tag{18}$$

where the log-likelihood is a representative measure of the model fit. Statistically, the higher the number, the better the fit. The preferred model, respectively the best combination of hyper parameters, is hence the one with the minimum AIC value.

In a second stage, the prediction performance of each algorithm on the test data splits is evaluated. In total, the three metrics discussed in IV-B are used as comparative values. Thereby we can investigate how well each individual algorithm performs both in terms of prediction accuracy and run-time efficiency. In the given context of HV current measurements, attaining a trade-off between run-time and accuracy is most critical. The focus of this stage of the analysis is thereby on identifying the algorithm that ensures the best trade-off. To this purpose, the optimal number of historical data points necessary per algorithm is also taken into consideration. The results of each algorithm relative to a prediction horizon of 20 time steps are provided in section V.

2) Bootstrap Aggregation Performance Assessment: Another core aspect of our work is to analyze the potential added value of Bootstrap Aggregation. Our hypothesis is that combining predictions derived from substantially different methods helps to increase the overall accuracy. As discussed in III-B, Bootstrap Aggregation is especially relevant in the present case given that the uncertainty about which method is most accurate and under which conditions the power train is

operating are high. We adopt formal procedures in combining predictions of the algorithms. In the ideal case, prediction errors are negatively related so that they might cancel each other. Thereby, we follow an equal-weights approach as described in the following equation

$$\hat{y}_{t+1} = \frac{1}{M} \sum_{i=1}^{M} \hat{y}_{t+1}^{(i)}, \tag{19}$$

where $\hat{y}_{t+1}^{(i)}$ is the predicted value of algorithm $i \in [1..M]$ at time step $t+1$ and $M = 7$ is the number of algorithms in the pool.

V. RESULTS

The focus of this section is on discussing the results obtained for the experiments described in section IV-D. Each experiment is discussed separately.

A. Individual Performance Assessment

In this subsection, we aim to assess the isolated performance of each prediction algorithm. To this end, we refer to the performance metrics discussed in subsection IV-B. The values obtained for the RMSE and MAPE metrics are summarized in Table I. For the comparison of the retrieved prediction performances, we use the naive method as benchmark. It is only outperformed by the ARIMA algorithm in matters of the MAPE. The MAPE improvement achieved by ARIMA is 8 %. These superior results are achieved due to the integration part of ARIMA. Thus, the algorithm can better adjust itself to our non-stationary data and retrieves only small outliers in its predictions. It is also relevant to mention that all considered algorithms except for BATS and TBATS achieved an average prediction error rate of less than 5 %.

However, the comparison with regard to the RMSE retrieves better results. ARIMA, BATS and TBATS outperform the benchmark. Especially BATS and TBATS achieve much better results. Their good RMSE results are interesting, because they both obtained relatively high MAPE values with respectively 7.47 % and 7.48 %. The other naive method, Random Walk, and the Exponential Smoothing Algorithms achieve comparable results like the benchmark. The discrepancies obtained both in terms of RMSE and MAPE hint that combining the forecasts of several algorithms might result in an improvement of the overall performance.

TABLE I
OVERVIEW OF THE PREDICTION PERFORMANCE SCORES OBTAINED BY THE ALGORITHMS ON THE TEST SET.

Algorithm	RMSE	MAPE (%)	% improvement over benchmark
Simple Exp. Smoothing	1,85	4,25	-0.2
Holt-Winter Exp. Smoothing	1.84	4.24	0
ARIMA	1.59	3.90	+8.0
TBATS	0.74	7.48	-76.4
BATS	0.75	7.47	-76.1
Random Walk	1.81	4.38	-3.3
Naive (Benchmark)	1.84	4.24	0

Table II extends the results described in Table I to cover computational aspects. As discussed in IV-B.3, computational complexity and run-time requirements are critical to our objective of predicting HV current measurements. Considering the prediction horizon of below 20 time steps, the methods of seasonal ARIMA, BATS and TBATS become obsolete. Their respective run-time exceeds our limit of at maximum $200\,\mathrm{ms}$. Under real usage conditions, the obtained forecasts would be irrelevant by the time they are computed. ARIMA misses the run-time limit only shortly. Further optimizations could make this algorithm feasible for our purposes. Without further optimizations, the focus should be set on the remaining simplest methods for deployment.

TABLE II
COMPARISON PER ALGORITHM WITH REGARD TO ACCURACY AND EFFICIENCY.

Algorithm	MAPE (%)	Run-Time (ms)	# Historical Data Points
Simple Exp. Smoothing	4.25	6	10
Holt-Winter Exp. Smoothing	4.24	7	10
ARIMA	3.90	490	40
TBATS	7.48	$18 \cdot 10^3$	10
BATS	7.47	$12 \cdot 10^3$	10
Random Walk	4.38	0.5	5
Naive (Benchmark)	4.24	0.064	1

B. Bootstrap Aggregation Performance Assessment

In this section, we discuss the results obtained during the experiment introduced in IV-D.2. For this experiment, we combine the 5 algorithms with the lowest MAPE. Thus, the algorithms considered for the combination are ARIMA, the simple and Holt-Winters Exponential Smoothing, as well as the Random Walk and the naive method, as can be seen from Table I. As mentioned in IV-D.2, we follow an equal-weights approach. The prediction values of each algorithm are therefore averaged at each prediction run-through. The hereby discussed results are obtained for the same data sets used in V-A. This enables an objective comparison of the individual and combined performances on the same data. For the same test set, the bootstrap aggregation resulted in an RMSE value of 1.74 and a MAPE value of 4.19 %. This means again a prediction error rate of below 5 %. Compared to the individual performances, the bootstrap aggregation outperformed all individual methods in terms of the MAPE except for ARIMA. Unfortunately, this contradicts the hypothesis discussed IV-D.2 that the combination of several algorithms is able to improve the overall prediction. Nevertheless, we think that future work can improve the here retrieved results with an adaptive weights approach. Instead of the equal weights approach used in the context of this work, the adaptive weights approach might be able to benefit from the high accuracy of ARIMA.

As expected, Bootstrap Aggregation has the worst run-time of all considered approaches. As the concept combines the predictions of several algorithms, it also sums up the required run-time of all these algorithms. Especially in our field of application, the execution on ECUs, the long run-

time is a heavy disadvantage. Regrettably, this disadvantage cannot be equalized by the here achieved results.

VI. CONCLUSIONS

Time delays between distributed systems lead to outdated measurement signals. But up-to-date input data is required for control functions, especially in real-time systems like power trains of EVs. A solution to this problem is to predict delayed signals until the present. The goal of this paper is to evaluate which algorithms are suited for time series prediction of delayed measurement signals of power trains of EVs. For this purpose, we evaluate 5 state-of-the-art time series prediction algorithms and 2 naive methods. As it is important for real-time systems to retrieve information in the required time frame, we focus our evaluation not only on accuracy which we measure with the RMSE and the MAPE, but also on the required run-time to execute the prediction. BATS and TBATS are the most accurate algorithms. However, due to their high outliers, they are unsuited for our purposes. ARIMA offers the best compromise between high accuracy and small outliers. As expected, the naive method is the fastest method. Surprisingly, although it is the simplest of all methods, its accuracy is not far below the other methods. Its relatively good results show the difficulty of predicting HV measurements of electric power trains accurately. Thus, further work needs to be done to enable fast and accurate predictions. A possibility for future work is to optimize ARIMA and try to make it faster. Another possibility is to combine the predictions of several algorithms with Bootstrap Aggregation. Although the here implemented equal weights approach outperforms almost all algorithms, it is not able to achieve the low MAPE value of ARIMA. Further work is necessary to investigate, if an adaptive weights approach is able to outperform ARIMA. Until now, Bootstrap Aggregation requires by far the most run-time. To make it feasible for the execution on automotive ECUs, further research to increase its efficiency is required.

REFERENCES

[1] J. Pfeiffer, X. Wu, and A. Ayadi, "Evaluation of Three Different Approaches for Automated Time Delay Estimation for Distributed Sensor Systems of Electric Vehicles," *Sensors*, vol. 20, no. 2, p. 351, Jan. 2020. [Online]. Available: https://www.mdpi.com/1424-8220/20/2/351

[2] J. Pfeiffer and X. Wu, "Automated Time Delay Estimation for Distributed Sensor Systems of Electric Vehicles," in *2019 10th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS)*. Metz, France: IEEE, Sep. 2019, pp. 609–614. [Online]. Available: https://ieeexplore.ieee.org/document/8924330/

[3] S. Makridakis, E. Spiliotis, and V. Assimakopoulos, "The M4 Competition: 100,000 time series and 61 forecasting methods," *International Journal of Forecasting*, vol. 36, no. 1, pp. 54–74, 2019. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0169207019301128

[4] A. Styler, G. Podnar, P. Dille, M. Duescher, C. Bartley, and I. Nourbakhsh, "Active management of a heterogeneous energy store for electric vehicles," in *2011 IEEE Forum on Integrated and Sustainable Transportation Systems*. Vienna, Austria: IEEE, Jun. 2011, pp. 20–25. [Online]. Available: http://ieeexplore.ieee.org/document/5973650/

[5] M. Ibrahim, G. Wimmer, S. Jemei, and D. Hissel, "Energy management for a fuel cell hybrid electrical vehicle," in *IECON 2014 - 40th Annual Conference of the IEEE Industrial Electronics Society*. Dallas, TX, USA: IEEE, Oct. 2014, pp. 3955–3961. [Online]. Available: http://ieeexplore.ieee.org/document/7049092/

[6] A. Bolovinou, I. Bakas, A. Amditis, F. Mastrandrea, and W. Vinciotti, "Online prediction of an electric vehicle remaining range based on regression analysis," in *2014 IEEE International Electric Vehicle Conference (IEVC)*. Florence: IEEE, Dec. 2014, pp. 1–8. [Online]. Available: https://ieeexplore.ieee.org/document/7056167/

[7] E. Zulkas, E. Artemciukas, D. Dzemydiene, and E. Guseinoviene, "Energy consumption prediction methods for embedded systems," in *2015 Tenth International Conference on Ecological Vehicles and Renewable Energies (EVER)*. Monte Carlo: IEEE, Mar. 2015, pp. 1–5. [Online]. Available: http://ieeexplore.ieee.org/document/7112932/

[8] F. Collopy and J. S. Armstrong, "Rule-Based Forecasting: Development and Validation of an Expert Systems Approach to Combining Time Series Extrapolations," *Management Science*, p. 25, 1992. [Online]. Available: http://www.forecastingprinciples.com/paperpdf/Rule-based%20Forecasting%20Development%20and%20Validation.pdf

[9] C. Shah, "Model selection in univariate time series forecasting using discriminant analysis," *International Journal of Forecasting*, vol. 13, no. 4, pp. 489–500, Dec. 1997. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0169207097000319

[10] B. D. Fulcher, "Feature-based time-series analysis," *arXiv:1709.08055 [cs]*, Oct. 2017, arXiv: 1709.08055. [Online]. Available: http://arxiv.org/abs/1709.08055

[11] N. Hatami, Y. Gavet, and J. Debayle, "Classification of Time-Series Images Using Deep Convolutional Neural Networks," *arXiv:1710.00886 [cs]*, Oct. 2017, arXiv: 1710.00886. [Online]. Available: http://arxiv.org/abs/1710.00886

[12] Z. Wang and T. Oates, "Encoding Time Series as Images for Visual Inspection and Classification Using Tiled Convolutional Neural Networks," in *Trajectory-Based Behavior Analytics: Papers from the 2015 AAAI Workshop*, Austin, Texas, USA, 2015, p. 7.

[13] R. G. Brown and A. D. Little, "EXPONENTIAL SMOOTHING FOR PREDICTING DEMAND," Philip Morris Records, Cambridge 42, Massachusetts, Tech. Rep., Nov. 1956. [Online]. Available: https://www.industrydocuments.ucsf.edu/docs/jzlc0130

[14] R. J. Hyndman and Y. Khandakar, "Automatic Time Series Forecasting: The **forecast** Package for *R*," *Journal of Statistical Software*, vol. 27, no. 3, 2008. [Online]. Available: http://www.jstatsoft.org/v27/i03/

[15] V. Kotu and B. Deshpande, *Data Science: Concepts and Practice*. Cambridge, MA, United States: Elsevier, 2019. [Online]. Available: http://asolanki.co.in/wp-content/uploads/2019/04/Data-Science-Concepts-and-Practice-2nd-Edition-3.pdf

[16] A. M. De Livera, R. J. Hyndman, and R. D. Snyder, "Forecasting Time Series With Complex Seasonal Patterns Using Exponential Smoothing," *Journal of the American Statistical Association*, vol. 106, no. 496, pp. 1513–1527, Dec. 2011. [Online]. Available: http://www.tandfonline.com/doi/abs/10.1198/jasa.2011.tm09771

[17] O. Gjølberg and B.-A. Bengtsson, "Forecasting quarterly hog prices: Simple autoregressive models vs. naive predictions," *Agribusiness*, vol. 13, no. 6, pp. 673–679, Nov. 1997. [Online]. Available: http://doi.wiley.com/10.1002/%28SICI%291520-6297%28199711/12%2913%3A6%3C673%3A%3AAID-AGR11%3E3.0.CO%3B2-1

[18] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, 2nd ed., ser. Springer Series in Statistics. New York, NY, USA: Springer, 2009.

[19] C. Bergmeir, R. J. Hyndman, and J. M. Benítez, "Bagging exponential smoothing methods using STL decomposition and Box–Cox transformation," *International Journal of Forecasting*, vol. 32, no. 2, pp. 303–312, Apr. 2016. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0169207015001120

[20] L. J. Tashman, "Out-of-sample tests of forecasting accuracy: an analysis and review," *International Journal of Forecasting*, vol. 16, no. 4, pp. 437–450, Oct. 2000. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0169207000000650

[21] C. Bergmeir and J. Bentez, "On the use of cross-validation for time series predictor evaluation," *Information Sciences*, vol. 191, p. 192213, 05 2012.

4 Deviation Correction

After correcting time delays in the previous chapter, this chapter deals with the correction of deviations caused by measurement inaccuracies. Section 4.1 shows how to detect such measurement deviations. After detection, the deviations can be corrected as shown in section 4.2.

4.1 Measurement Deviation Detection

Following Hypothesis H1 in section 1.3, this paper focuses on deviations between real and measured currents caused by measurement inaccuracies.

It validates Hypothesis H2b by learning a measurement model on-board of each vehicle in a fleet of equally constructed EVs. The EVs send the measurement models to a back-end, once the models are trained. In the back-end, all models are applied to a unique input data set, the so-called *unity drive*. Afterwards, the outputs on the unity drive of all models are averaged. The average is assumed to be the real value without measurement deviations, the *ground truth*. The deviation between each EV and the fleet serves as input for our classifier. By taking additionally the measurement deviation between the HVB and the considered HV component of the EV into account, the classifier is able to detect measurement faults as well as hardware faults. The fault values determined by the classifier can serve as input for a measurement correction in the next development stage (see section 4.2).

By training only the measurement model on-board and outsourcing the ground truth estimation to a back-end, this approach validates Hypothesis H3. Clearly, it is more efficient to transmit the measurement model parameters and the mean deviation between HV component and HVB to the back-end only from time to time than to transfer the raw data completely during each time step. To enable the relatively expensive training of the measurement model on ECUs, the paper suggests to collect some raw data on-board during driving and execute the training once the power train ECU has the available resources free for a sufficiently long time period, like for example during charging.

Hypothesis H4 is validated by the proposed algorithms in the paper. The unsupervised training of the measurement model works completely without manual parametrization. This means that there is no work for the applicator here. However, the classifier requires in its current stage that the decision boundaries are set manually.

This would mean that there are two new parameters which need to be adjusted by an applicator.

My own contribution to the paper is the basic idea for the fleet-based approach as well as the idea to take subspace identification algorithms like Multivariable Output Error State Space (MOESP) and Numerical State Space Subspace System Identification (N4SID) into account for learning the measurement model and to compare their performance to Neural Networks. The final versions of the proposed fleet-based approach and the classifier are the result of two joint brainstorming sessions with Peter Wolf, Roberto Pereira and me. Peter Wolf is responsible for the implementation and training of the LSTM and the classifier used in the scope of this paper. Although I implement the very first versions of the MOESP and N4SID algorithms, it is Roberto Pereira's decision to focus on the N4SID algorithm in the context of this paper. The final version of the implementation is also his work. I execute all the test drives for recording the training data for the models. Furthermore, I am responsible for the introduction, the research into the state of the art and the conclusion written in the paper. Beyond these three chapters, I review and edit all parts written by the other two authors.

2019 IEEE Intelligent Vehicles Symposium (IV)
Paris, France. June 9-12, 2019

A Fleet-Based Machine Learning Approach for Automatic Detection of Deviations between Measurements and Reality

Jakob Pfeiffer[1,2], Peter Wolf[1,3] and Roberto Pereira[1,2]

***Abstract*— Deviations between system current measurements and reality can cause severe problems in the power train of electric vehicles (EVs). Among others, these are inaccurate performance coordination and unnecessary power limitations during driving or charging. In this work, we propose a fleet-based framework to detect such deviations. Our main assumption is that the real value is the mean of all identically constructed EVs' measurements for the same input. Under this assumption, we train individual on-board models to predict the current of the electric machine (EM) and transmit the model parameters to a back-end. There, we compare individual deviations of the predicted current against the fleet in the same scenario. We use the results to classify three fault sources. As models we choose two different Machine Learning algorithms: State Models and Long Short-Term Memory Neural Networks (LSTMs). These are evaluated on an artificial fleet of 34 EVs derived from real drive data containing three different kinds of faults. Results show that our proposed approach correctly classifies major measurement faults. Additionally, both models offer similar classifications. LSTMs are more accurate, whereas state models are less computationally complex, and thus better suited for electronic control units (ECUs).**

I. INTRODUCTION

According to *Kirchhoff's current law*, the sum of all currents in an electric system is equal to 0. Kirchhoff's theory might match the current measurements in high voltage (HV) systems of academic or prototype vehicles with expensive and extremely accurate sensors. However, considering measured signals of high volume EVs with off-the-shelf sensors, the sum of all currents can differ up to $20\,\%$ of the maximum current (see Fig. 1). More detailed, the root mean square error (RMSE) of the sum of currents $\mathrm{RMSE}(i_{\mathrm{sum}}) = 0,67\%$ is as high as the average current of the DCDC converter $\mu_{i_{\mathrm{DCDC}}} = 0,67\%$ (see the zoomed section of Fig. 1). Even its variance $\sigma_{i_{\mathrm{sum}}} = 1,88\%$, which indicates the uncertainty of all measurements, is higher than the consumption of smaller components. It reaches approximately half of the consumption of the second largest consumer $\mu_{i_{\mathrm{heat}}} = 3,78\ \%$.

The divergence between measurements and reality becomes problematic when the system is operating close to its boundaries. To increase safety and ensure long battery lifetimes, the amount of power charged to or discharged from the high voltage battery (HVB) needs to be restricted [1]. In addition, to guarantee a safe operation mode even under worst measurement conditions, the maximum sensor inaccuracy is added as an offset to this restriction. However, this offset comes with drawbacks during discharging and charging. In the former case, the system might not release requested power, although the HVB could provide it in reality. In the charging case, especially during recuperation, the HVB might not allow the full power level, although it would be capable to handle it. This results in a restricted performance and cruising range of the EV.

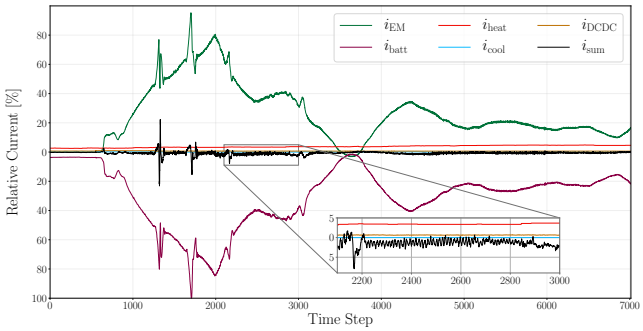


Fig. 1. Currents of all HV components in an EV on a test drive. The sum of all currents i_{sum}, which should be constantly 0 according to Kirchhoff's current law, is plotted in black. Zooming into the measurements shows that the deviation i_{sum} is higher than the current of the DCDC converter i_{DCDC}. Even its noise spectrum is approximately half as high as the consumption of the heating i_{heat}, which is the second largest consumer in this drive.

The aim of this work is to automatically detect the deviation between sensor measurements and real values. We use a fleet-based approach to identify deviant sensors and offer a basis for measurement correction. Besides enhancing a more accurate power distribution, this allows to reduce the battery protection offset and thus increase the performance and range of EVs.

The rest of this paper is structured as follows. Section II states related work and our contributions. In section III, we explain the theory behind our work before elaborating on the circumstances of our experiments in section IV. The results of these experiments are discussed in section V. Finally, we draw our conclusions in section VI.

II. STATE OF THE ART

For this work, we distinguish between two different kinds of faults: *measurement* and *hardware faults*. The first describes faulty measurement data caused by, e.g., corrupted sensors. The second means that the sensors correctly measure wrong behaving hardware, e.g., defect actuators. Furthermore, we distinguish between two groups of approaches for measurement correction with machine learning methods.

[1] BMW Group, Petuelring 130, 80788 Munich, Germany {Jakob.J.Pfeiffer, Peter.WP.Wolf, Roberto.Pereira}@bmwgroup.com
[2]TU Munich, Data Processing, Arcisstr. 21, 80333 Munich, Germany
[3]TU Dresden, Vehicle Mechatronics, Helmholtzstr. 10, 01069 Dresden, Germany

The first group are off-board approaches and the second one is composed of on-board approaches. Off-board approaches train a model outside of its measurement environment. The training is based on simulation or previously recorded data. The model is executed on-board and deviations from a pre-trained behavior can be detected.

Malakar et al. [2] use an off-board approach with Neural Networks to increase the quality of their measurements. It detects input signals which lead to a bias in the measurement output. To drop the bias, Malakar et al. neglect these signals. However, even if a signal, or the model that created it is corrupted, it still might contain parts which carry valuable information [3]. Unfortunately, neglecting the whole signal also drops valid parts of the information. Therefore, we prefer a correction of the signal to its cancellation. Another difference between Malakar's approach and ours is that their measurement environment consists only of the sun and the air, which means that deviations in their data are always measurement faults and not hardware faults.

An example for the determination of hardware and measurement faults is shown by Zhao et al. [4]. They build a simulation model of an aero-engine. It contains sub-models for all components including actuators and sensors which are implemented based on physical principles. With the help of principle component analysis (PCA) and diagnosis models, Zhao et al. can detect deviations between expected and measured values. Due to the component-wise modeling, they can draw conclusions about the occurrence of sensor faults. Their modeling technique demands expert knowledge about the physical principles of the measurement environment. If physical modeling is not carried out accurately enough, a problem might occur which is called the *reality gap* in the fields of evolutionary robotics [5]. It describes the phenomena that models often perform well in simulations, but fail when they are confronted with the real world during execution [3]. The reasons for this failure are that training data samples are often only available in certain working conditions, whereas the environment of the system varies across a broad range during execution [6].

In general, the reality gap must be considered when working with off-board approaches. Usually, training of machine learning methods is quite time-consuming [3]. The advantage of off-board approaches is that they separate the expensive training from a low-performance execution platform. This makes the use of higher computation and memory resources available for training and enables to apply a broad range of algorithms for analyzing the input data. Furthermore, the training does not need to take place in real-time. However, the drawback of these approaches is that situations might occur during execution which the training did not cover. These algorithms are not able to adapt to new circumstances and thus perform suboptimally [6].

In the second group of on-board approaches, the models are directly trained on the execution platform and continuously updated.

Jo et al. [7] show a particle filter based correction of bias errors of GPS sensors on-board of an experimental vehicle. As a series vehicle's standard hardware is not sufficient for their approach, they require additional sensors.

The sensor set of a series production engine is sufficient for the inspiring approach of Lu et al. [6]. They introduce a new on-board approach with an Extreme Learning Machine for sensor fault detection and apply it to the control system of an aero-engine. It is capable to distinguish between drift and bias faults and provides corresponding compensation data. Unfortunately, their approach is not able to distinguish between hardware and measurement faults.

These faults can be differentiated by Kobayashi and Simon [8]. Similar to Lu et al., their goal is to detect faults in an aero-engine as well. For that purpose, Kobayashi and Simon use a bank of Kalman Filters. Each sensor signal is monitored by a separate filter. An additional signal is designed to detect hardware faults. Based on a decision matrix, corrupted signals are isolated. However, especially for high dimensional problems, a separate filter for each signal results in a high number of filters and thus high computational costs.

On one hand, it is a general advantage that on-board approaches continuously update their model. Therefore, these algorithms can adapt to never before experienced situations. On the other hand, on-board learning suffers from three main disadvantages. First, especially in the automotive domain, cost effective design mostly prevents to add additional memory and performance to ECUs. This restricts the on-board learning capability and makes many algorithms infeasible. Second, the training is often required to be executed in real-time. Third, on-board approaches detect deviations only from otherwise working sensors. If the sensor returns biased measurements from the beginning, the data is mistakenly assumed to be correct.

Our contribution is the development of a fleet-based approach with an on-board trained measurement model and an off-board fault classifier. To the best of our knowledge, this paper is the first to propose such a hybrid measurement deviation detection approach for close-to-production EVs. Thus, we combine the ability to handle unseen situations with detecting ab initio corrupted sensors. We efficiently minimize data volume to be transferred over the air while being able to use the resources of a back-end. We evaluate our approach on real world data from the power train of a series EV without additional sensors. For building the on-board measurement model, we present and compare two approaches: State Models and LSTMs. Finally, the proposed classifier is capable to differentiate between hardware and two kinds of measurement faults.

III. CONCEPT

In this section, we explain the process of our measurement deviation detection approach first. Then, we present the theory used for model training before introducing our classifier.

To identify the deviation between measurements and reality for each vehicle, we propose a concept comprised of an on-board and off-board unit as plotted in Fig. 2. The main challenge is to find the real value, the so-called *ground truth*.

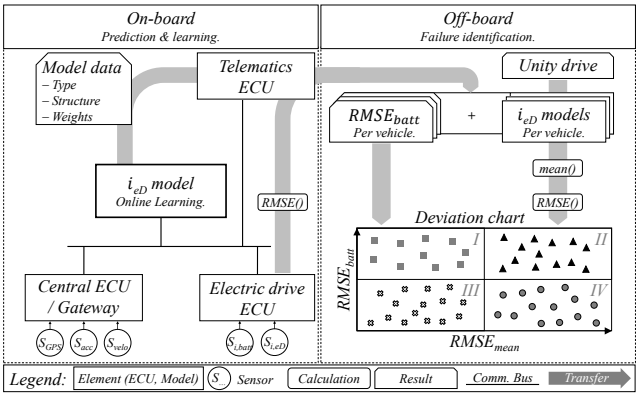


Fig. 2. Our concept for retrieving each vehicle's measurement deviation. We compare the measurements of the EM to the HVB and the fleet to determine if potential deviations are caused by inaccurate measurements or by hardware failures.

Due to the cost-effective design of high volume EVs, the ground truth cannot be retrieved on-board from redundant current sensor systems. An alternative way is to get the ground truth from other EVs. For a sufficiently high number of vehicles, the fleet's mean tends towards the true value apart from systematic errors. However, our goal is not to correct systematic errors of the whole fleet, but to minimize the individual deviation from the ground truth for each EV. Thus, we assume the ground truth to be the mean of a fleet containing as many EVs as possible of the same type.

A. Process

Our process for retrieving the ground truth and its deviation to the vehicle's measurements is visualized in Fig. 2. In the following, we exemplify our approach with measurements of the current of the EM. Nevertheless, the same approach can be transferred to other HV components of EVs.

Transferring all measurements from the EVs to the back-end is unfeasible. The amount of data would be too high for the available bandwidth. Therefore, we first collect data during driving. This can be optimized by taking account of the planned route (e.g., with information from the navigation system) and only store drives with an expected high variety of data (see IV-A). When we collected enough data, we build a simulation model on-board of the EV which returns its (biased) EM current measurements. Since the training algorithms require relatively high resources, we recommend to execute the training when the power train ECU is not busy and available for a long time period, e. g. during charging. After training, the measurement model as well as the on-board calculated deviation between the EM and the HVB (calculated as RMSE) are transferred to a back-end which collects all models of the considered fleet.

In the back-end, the ground truth and deviation determination take place. As stated above, we assume the ground truth to be the mean of all measurement models of the same fleet. Since it is not feasible to just calculate a mean over all collected models [9], it is retrieved in the same scenario using a *unity drive*. The unity drive is an input dataset from a real drive. The corresponding output is obtained by executing each measurement model on the unity drive. The mean of all outputs of the whole fleet is then assumed to be the true, unbiased measurement.

Once we have the ground truth, we compute the RMSE against the current of the EM. We compare this RMSE with the one we calculated on-board before. Based on these RMSEs, our classifier returns the scale as well the type of the deviation (see III-C).

B. Algorithms

As previously discussed, we compare two classes of algorithms: State Models and LSTMs. The first one is chosen due to its capability of modeling dynamic systems with a small amount of parameters. These parameters can be learned efficiently, e.g. with the State Space Subspace System Identification (N4SID) algorithm. Efficiency is a crucial feature for Embedded and Real-Time Systems. As alternative approach, we choose LSTMs due to their accurate learning of patterns in time [10]. We explain both methods here.

1) State Models: A discrete-time signal can be approximated by the state space model defined as

$$\begin{aligned} x(t+1) &= Ax(t) + Bu(t) + w_k \\ y(t) &= Cx(t) + Du(t) + v_k, \end{aligned} \tag{1}$$

where $x(t) \in R^{n_x}, u(t) \in R^{n_u}, y(t) \in R^{n_y}$ represent the model's state, input and output at time step t, respectively. Matrices A, B, C, D carry information about the system's behavior. The objective, at training time, is to identify those matrices given an observed sequence of input $u(t)$ and output $y(t)$ data.

There are many existing methods to extract parameters $\theta = \{A, B, C, D\}$ from data, e.g., [9], [11], [12]. In practice, these algorithms can be computed using a standardized approach [13].

In such approaches, matrices A and C can be extracted from the weighted transformation of the observability matrix

$$\hat{O} = W_1^{-1} U_1 S_1, \tag{2}$$

where S_1 represents first n_x most relevant singular values of the observability matrix $\hat{O}$ and U_1 its singular vectors. The weighting matrix W_1 is defined according to [13].

After finding A and C, approximating B and D [9] can be done by solving the least square problem

$$\begin{bmatrix} X_{t+1} \\ Y_t \end{bmatrix} = \begin{bmatrix} A & B \\ C & D \end{bmatrix} \begin{bmatrix} X_t \\ U_t \end{bmatrix}. \tag{3}$$

Recursive solutions, which do not directly regress B and D, usually work best only with slowly changing signals [14], [15]. This is not our case, as the required EM current can rapidly vary from the lowest to the highest signal point in a short period of time.

The problem of finding a recursive implementation of subspace models that works well on fast changing signals is a big disadvantage compared to LSTMs. However, especially for our problem, we argue that subspace models require less data and computational power. The learning step of Numerical Algorithms for N4SID, a special type of state

models [16], consists of data projection and decomposition [13], which makes it a suitable method for our approach.

2) Long Short-Term Memory Neural Network (LSTM): Another promising approach to regression on high-frequent time series in the field of ML are LSTMs. This special type of recurrent neural network was proposed by Hochreiter and Schmidhuber [17] and further improved by Gers et al. [10]. By using the concept of gating, LSTMs have the ability to remember inputs for a long time. Approaches using LSTMs have already achieved state-of-the-art results in speech recognition [18] or human activity recognition [19]. Additionally, this network type has successfully been applied in a stacked architecture to similar ECU data to detect faults in a combustion engine [20]. For more detailed information about LSTMs, the reader is kindly referred to [17] and [10].

The capabilities of LSTMs are used in our second model to directly map present and past vehicle motion information to the actual current required at the EM. For that, we build a network which is comprised of 2 LSTM layers with 7 neurons each and a dense output layer with 1 neuron. The first 2 layers extract relevant information from the features and the last neuron transforms these information to the actual current. As activation functions for the (output) hidden state and inner recurrent activation, *tanh* and *hard sigmoid*, and *softmax*, respectively, are selected.

The training is done in a supervised manner backpropagating gradients from the output layer to the first LSTM layer. The network parameters are optimized using stochastic gradient descent to minimize the mean squared error loss with Adam [21]. This optimizer provided the most stable results in our experiments. Since we do offline batch training in the vehicle, this procedure is executed after a predefined amount of new data is collected. Our LSTM model contains 1100 network parameters trained in 40 epochs using a batch size of 2048. The inputs are obtained using a sliding window approach (see IV-A) with a window length of 25 time steps and a step size of 5 time steps. A learning rate and decay factor of $20e^{-4}$ and $10e^{-5}$, respectively, are applied. Weight initialization is conducted orthogonally for all layers. This setup is obtained through an optimization (see V-C) and leads to the best results for this application.

C. Classifier

Our classifier is built upon comparison rules. To identify sensor faults, we compare $\text{RMSE}_{\text{batt}}$ and $\text{RMSE}_{\text{mean}}$. $\text{RMSE}_{\text{batt}}$ is calculated on-board between the current of the EM and the HVB. The off-board calculated $\text{RMSE}_{\text{mean}}$ denotes the deviation between the current of the individual EM and the mean current of the fleet. The final fault classification is done with the deviation chart shown in Fig. 3.

The decisions in the chart are based on the following considerations. If the deviation between an individual vehicle and the fleet is high ($\text{RMSE}_{\text{mean}}$ high), the vehicle behaves differently than the others. The reason for this might be either a hardware or a measurement fault. However, by looking at $\text{RMSE}_{\text{mean}}$ alone, we are not able to distinguish between the two cases. On the other hand, if we only look at $\text{RMSE}_{\text{batt}}$,

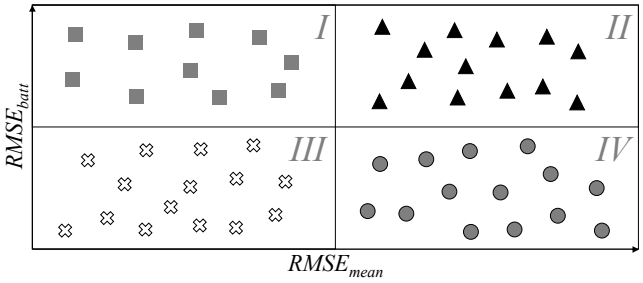


Fig. 3. Deviation chart for classifying measurement deviations. Group I represents EVs with an HVB measurement fault. EVs with a measurement fault in the EM belong to group II. Group III contains healthy EVs without recognizable faults and group IV consists of EVs with a hardware fault.

we can state if there is a measurement fault, but not where it occurs. The combination of both values provides us the needed information.

This results in the following four cases. First, consistency between the EM measurements and the fleet but not the HVB data (case I in Fig. 3) indicates a current measurement fault of the HVB. Second, additional inconsistency with the fleet denotes an EM measurement fault (case II). Third, no or small deviations between the EM measurements and the HVB as well as the fleet describe the ideal case when the measurements and the hardware work correctly (case III). Finally, if the measurements of the EM and the HVB are consistent, but the fleet behaves differently, a hardware fault is highly probable (case IV).

The boundaries separating the cases can be obtained by two procedures, i.e., data perturbation or maximum distance. Both procedures start with a healthy vehicle. For data perturbation, the boundaries are set between one vehicle known as healthy and a first perturbed dataset which predicts a current considered as *just about* faulty. When using maximum distance, the boundaries are set at a predefined maximum mean deviation (distance) which is considered as border of being not healthy anymore. The current and maximum mean deviation are initially defined by domain experts.

IV. EXPERIMENTAL SETUP

In this section, we further explain the data we use in this work as well as the setup and environment of our experiments for evaluating State Models and LSTMs.

A. Data

For training our models, we use real data recorded during representative drives on public roads. In the context of this paper, we use 1 real EV and generate an artificial fleet of 33 faulty EVs based on the real data set. As we expect the classification to become better with data from more EVs, the ideal data set would consist of all identically constructed EVs. Per vehicle, the training data allocates 87 MB of memory and corresponds to $2\,\text{h}\ 41\,\text{min}$ of driving. Lossless reduction of this data is still up to research. The output data is one signal and is learned by the models namely the *current* consumption of the EM, as we want to simulate this signal. In our experiments, we use data according to physical principles as input to the models. These are the vehicle's

velocity, *acceleration*, the selected *gear*, *requested power*, and the *number of rotations* of the EM. Further improvement is achieved by adding the *temperature* and the *altitude* of the vehicle's environment.

For the evaluation of the classification, we create an artificial fleet containing 1 healthy EV, 11 EVs with a measurement fault, 11 EVs with a hardware fault and 11 EVs with an HVB measurement fault. Following [6], the faults consist of drifts, offsets and additionally pulses, where we multiply or add a value or add random noise, respectively. We choose the values such that the healthy EV's measurements match their mean. The values of the perturbations are printed in Table I.

The input vectors for both models are obtained using a sliding window approach. A window of a fixed size is sliding over the input data with a predefined step size. Since we want to predict the current at every time step, a sliding window with a length of w and step size s results in an overlap of $w-s$ time steps. Additionally, the inputs of the LSTM model are scaled to an interval of $[-1, 1]$ for training and rescaled to the original scale for evaluation.

B. Experiments

We perform three experiments for the state model and LSTM approach. First, each learned model predicts the current of the EM on the mean unity drive per perturbation type. The results are passed to the classifier to identify the source of the measurement deviation (see V-A). Second, we measure the final performance of each approach against the ground truth (current of the EM) as an overall performance and per perturbation type. This experiment is done using the unseen test set and gives us insights on the prediction error (see V-B). Third, we conduct hyperparameter tuning to get the optimal measurement models for this work. The optimization is done using the validation dataset and is discussed in Sec. V-C.

C. Environment

The LSTM model is implemented in Tensorflow[1] using Keras[2], a high-level application programming interface to generate and train neural networks. The state model is implemented using the linear algebra functions from SciPy[3]. Training, validation and testing are executed on an HP^{TM}Z-840 with two Intel® Xeon® E5-2640 v4 2.4Ghz CPUs and 96GB DDR4 RAM. All models are implemented in Python 3.6 with Microsoft Windows 10 as operating system.

V. RESULTS AND EVALUATION

In this section, we present the results of our experiments and discuss both algorithms regarding the classification, model performance and the required hyperparameters.

[1]Martín Abadi et al. Tensorflow: Large-scale machine learning on heterogeneous systems, 2015.

[2]François Chollet et al. Keras. https://keras.io, 2015.

[3]Jones E, Oliphant E, Peterson P, et al. SciPy: Open Source Scientific Tools for Python, 2001

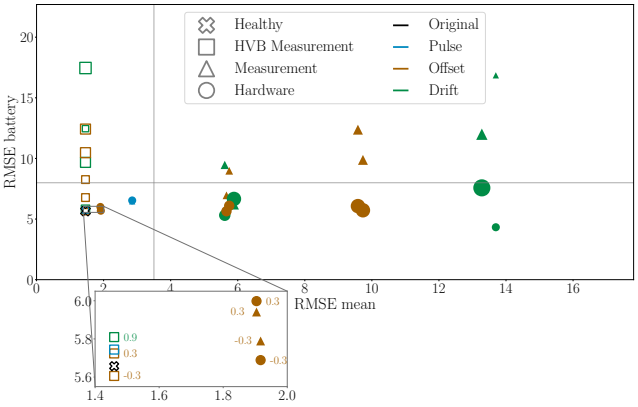


Fig. 4. The resulting classification of the LSTM approach. The size of the data point represents the perturbation value.

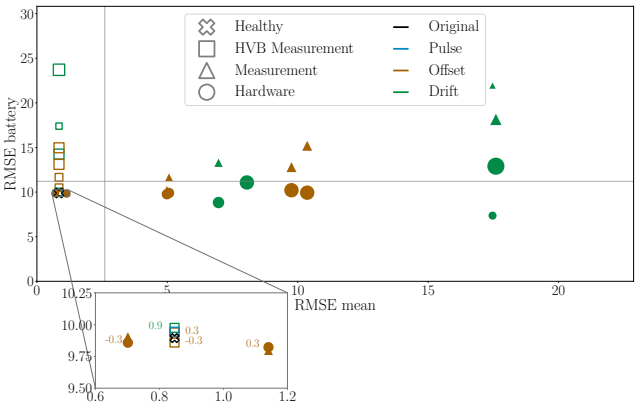


Fig. 5. The resulting classification of the State Model approach. The size of the data point represents the perturbation value.

A. Classifier results

We train both LSTMs and State Models on the artificial fleet of 34 EVs. The resulting classifications in Fig. 4 and Fig. 5 show similar outcomes for both of our approaches. Group I and II do not contain any false positives with one exception for N4SID. Both models misclassify EVs with very small fault values (e.g., offset faults of $\pm$ 0.3 A) as healthy. With increasing perturbation values, measurement and hardware faults drift apart. We observe that HVB measurement faults move away from the healthy section on an approximately vertical line with increasing perturbation values. This is due to the fact that these models learn the same EM current data. In contrast to that, all other models are trained on varying (perturbed) EM current. Hence, the deviation against the mean fleet and HVB current varies on a higher extent.

B. Model performance

Training the LSTM in our setup includes sources of randomness, i.e., weight initialization and data shuffling. Therefore, we train and test the LSTM model multiple times and calculate statistics of the RMSE (mean, standard deviation, maximum, minimum) to gain more insights about a realistic RMSE result over several experiments. In opposite to that, as described in section III-B.1, our N4SID implementation

consists of data decomposition and solving a least squares problem. Therefore, no randomness is inserted and the model is trained and tested once.

1) LSTM: The performance evaluation for LSTM is carried out for 240 runs in total and thus 20 runs per perturbation type. Each run includes training and testing where the mean training time amounts to $359.11\,\mathrm{s} \pm 5.09\,\mathrm{s}$. The RMSE and related statistics are calculated as performance measures on all datasets and per perturbation type.

As a result, the LSTM model achieves an RMSE of 5.15 ± 0.61 over all runs and perturbation types with a minimum of 3.86 ± 0.16 at a sensor drift of 0.75. The maximum deviation of prediction and ground truth results in a RMSE of 6.41 ± 0.24 at a sensor drift of 1.25. On the original dataset, the LSTM is capable of predicting the current with a mean RMSE of 5.10 ± 0.20 and a minimum of 4.80. Based on the results in Table I, we observe that higher drifts and a pulse lead to a higher error while more extreme offset values have less impact. Additionally, a comparison of the predictions and the ground truth show that deviations in sections of negative current are twice as high as in sections of positive current, e.g., on the non-perturbed dataset, we achieve RMSEs of 8.47 and 3.97, respectively.

The first aspect is due to the error types chosen in this work. A sensor drift amplifies an erroneous prediction on the original data and a pulse adds random noise which cannot be predicted based on the input data. An offset influences the starting point of the curve but not the shape of the waveform itself, which is less difficult to learn. The second aspect is due to the availability of features to the model. Right now, we only consider the requested power at the electrical engine. This signal is always equal to or larger than 0 since recuperated energy, e.g., during breaking, is not taken into account. Thus, the model is provided with less detailed information about upcoming negative power at the engine. This effect is further increased when applying a sensor drift to the data, which acts as an amplifier. An improvement might be achieved if further signals about the negative regions are included, such as recuperation level or, to some extent, the brake pedal position.

2) State Models: As discussed in section III-B.1, the difficulty in designing a recursive version of N4SID, which works on fast changing signals, leads us to use an offline implementation [13].

The first challenge is to simulate cars with multiple gears. Different gears imply different signal behavior and gear shifts insert perturbations to the signals. Both are hard to model using a single subspace model. To tackle this problem, we use one subspace model per gear. This results in a accumulated training time of $221.95\,\mathrm{s}$ for N4SID.

Following this approach, subspace models achieve an RMSE of 9.29 over all runs and datasets with a minimum of 6.84 at sensor drift of 0.75. The most misled prediction results in an RMSE of 11.91 and is generated by sensor drift of 1.25. The RMSE of the original dataset reaches an RMSE of 9.14 and is below the overall error.

When comparing these results with the LSTM, subspace models lack simulation capability. To improve the model, we insert auxiliary points during training and test time (see Sec. V-C.2 for further details). This reduces the overall RMSE to 5.90, the minimum to 5.11, and the maximum RMSE to 7.24. These values are shown in table I by N4SID(AP).

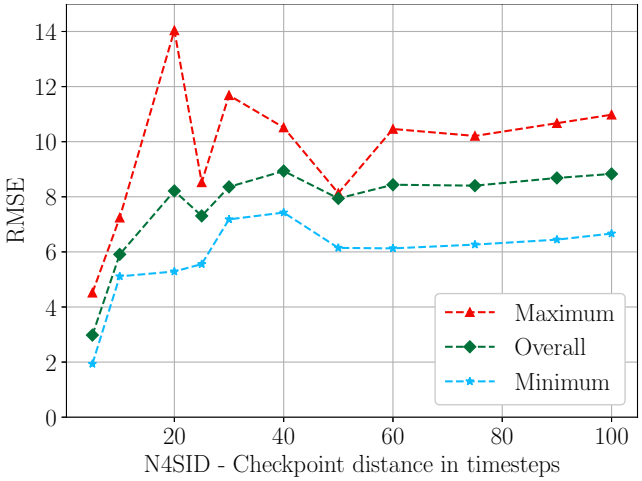


Fig. 6. Relationship between RMSE and checkpoint distance for all perturbation types. The overall, minimum, and maximum RMSEs increase with an increasing checkpoint distance.

Two signal properties found during the LSTM experiment can be observed in the subspace experiments as well. First, higher drifts and pulse lead to higher prediction errors. Second, predictions in positive areas of current are more accurate than in negative regions.

In summary, LSTMs are more accurate than State Models in the setup of this work and achieve an RMSE of 5.15 ± 0.61 compared to 5.90. However, lower requirements on memory and computation performance for training let State Models be better suited for an on-board application.

C. Hyperparameter

1) LSTM: Three hyperparameter sets are evaluated in order to gain insights on the most suitable architecture for the setup in this work: Sliding window length and step size, number of layers, and number of neurons per layer. We vary the sliding window length and step size in the value range of $\{5, 10, 15, 18, 20, 22, 25, 30, 35, 40, 45, 50\}$ and $\{1, 2, 3, 4, 5\}$ time steps, respectively, to determine the optimum of information provided to the model. The number of layers are varied in $\{1, 2, 3\}$ with a range of $\{1, 2, 3, 4, 5, 6, 7\}$ neurons per layer. This parameter set is evaluated to determine the optimal complexity of a function which maps the input signals to the actual current at the EM. The learning and decay rate are set to 20^{-4} and 10^{-5} so that the loss converges after approximately 40 epochs (determined by a grid search in $\{10^{-5}, 10^{-4}, 20^{-4}, 10^{-3}\}$ and $\{0, 10^{-5}, 10^{-4}, 10^{-3}\}$). Through an early removal of low performance configurations, we reduce the possible combinations so that approximately 800 model configurations in total are trained and tested.

2) State Models: Two hyperparameter sets are optimized for better performance of state models: zero-padding between gears and distance of auxiliary values.

TABLE I

$RMSE$ AND TRAINING TIME OF LSTM AND N4SID WITHOUT (N) AND WITH AUXILIARY POINTS (AP) ON THE TEST SET.

Perturbation		$RMSE$				
Type	Value	N4SID		LSTM		
	-	N	AP	$mean \pm std$	min	max
Origin	-	9.14	5.84	5.10 ± 0.20	4.80	5.58
Pulse	$\mathcal{N}(0,\,1)$	9.18	6.74	5.24 ± 0.33	4.77	6.09
Drift	0.75	6.84	5.11	$\mathbf{3.86 \pm 0.16}$	**3.54**	**4.20**
Drift	0.90	8.19	5.11	4.59 ± 0.19	4.33	5.07
Drift	1.10	10.43	7.24	5.57 ± 0.30	4.97	6.14
Drift	1.25	11.91	6.94	6.41 ± 0.24	6.04	7.07
Offset	-10.00	9.18	5.85	5.08 ± 0.18	4.75	5.38
Offset	-5.00	9.21	5.66	5.25 ± 0.26	4.81	5.88
Offset	-0.30	9.07	5.66	5.12 ± 0.19	4.79	5.47
Offset	0.30	9.18	5.29	5.17 ± 0.23	4.71	5.63
Offset	5.00	9.01	5.41	5.13 ± 0.29	4.76	5.83
Offset	10.00	9.30	5.52	5.22 ± 0.24	4.87	5.81
Overall	-	9.29	5.90	5.15 ± 0.61	3.54	7.07

The first describes the number of added zeros in between the gear shifts. This is required since a gear shift causes perturbation of data (e.g. peaks, noise). To avoid negative effects on model training, zeros are added to reduce the impact of perturbation.

The second hyperparameter refers to the insertion of checkpoints for training and validation. We observe in our experiments, that whenever there is a fast change in the current signal, subspace models usually fail to detect it. To overcome this problem, we use past ground truth data as checkpoints and enforce the correctness of the model. This has shown to be a useful trick to boost the method performance. As shown in Fig. 6, increasing the distance of auxiliary points increases the model error. To preserve memory and processing resources, we choose to insert the checkpoints every 10 time steps, representing a maximum increase of $10\,\%$ of storage.

In total, 192 experiments are conducted. We use value ranges of $\{0, 5, 10, 20, 25, 50, 100, 150\}$ and $\{0, 5, 10, 20, 25, 35, 45, 50, 60, 75, 90, 100\}$ for the amount of zero-padding and the distance of checkpoints, respectively. Each combination is executed for multiple and single gear approaches. The best balance between data storage and performance is acquired with zero-padding of 100 and inserting checkpoints every 10th time step. We use N4SID as subspace model implementation [13].

VI. CONCLUSION

This paper presents a fleet-based classification framework to distinguish between hardware and measurement faults in EV power trains. We combine on-board training and off-board classification to identify individual EV faults through a comparison with the mean behavior of the fleet. Two different algorithms, LSTMs and State Models, are proposed to offer a more accurate and more resource effective solution, respectively. We choose a data efficient approach by transferring the on-board models only instead of extensive vehicle data. The application on an artificial fleet derived from real world data shows that our classifier is capable to distinguish between major hardware, EM and HVB measurement faults. This serves as basis for measurement corrections to increase the performance as well as the efficiency of EV power trains.

REFERENCES

[1] P. Komarnicki, J. Haubrock, and Z. A. Styczynski, "Electrical components of an EV (Elektrische Komponenten des E-Kfz)," in *Elektromobilität und Sektorenkopplung*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2018, pp. 61–109.

[2] N. K. Malakar, D. J. Lary, A. Moore, D. Gencaga, B. Roscoe, A. Albayrak, and J. Wei, "Estimation and bias correction of aerosol abundance using data-driven machine learning and remote sensing," in *IEEE Conference on Intelligent Data Understanding (CIDU)*, 2012, pp. 24–30.

[3] S. Koos, J.-B. Mouret, and S. Doncieux, "The transferability approach: Crossing the reality gap in evolutionary robotics," *IEEE Transactions on Evolutionary Computation*, vol. 17, no. 1, pp. 122–145, 2013.

[4] Z. Zhao, Y.-g. Sun, and J. Zhang, "Fault detection and diagnosis for sensor in an aero-engine system," in *Chinese Control and Decision Conference (CCDC)*. IEEE, 2016, pp. 2977–2982.

[5] N. Jakobi, P. Husbands, and I. Harvey, "Noise and the reality gap: The use of simulation in evolutionary robotics," in *Third European Conference on Artificial Life*, vol. 929, 1995, pp. 704–720.

[6] J. Lu, J. Huang, and F. Lu, "Sensor fault diagnosis for aero engine based on online sequential extreme learning machine with memory principle," *Energies*, vol. 10, no. 1, p. 39, 2017.

[7] K. Jo, K. Chu, and M. Sunwoo, "GPS-bias correction for precise localization of autonomous vehicles," in *IEEE Intelligent Vehicles Symposium (IV)*, 2013, pp. 636–641.

[8] T. Kobayashi and D. L. Simon, "Evaluation of an enhanced bank of kalman filters for in-flight aircraft engine sensor fault diagnostics," *Journal of Engineering for Gas Turbines and Power*, vol. 127, no. 3, p. 497, 2005.

[9] M. Verhaegen, "Identification of the deterministic part of mimo state space models given in innovations form from input-output data," *Automatica*, vol. 30, no. 1, pp. 61–74, 1994.

[10] F. A. Gers, J. Schmidhuber, and F. Cummins, "Learning to forget: Continual prediction with lstm," *Neural Computation*, vol. 12, no. 10, pp. 2451–2471, 2000.

[11] P. van Overschee and B. de Moor, "N4sid: Subspace algorithms for the identification of combined deterministic-stochastic systems," *Automatica*, vol. 30, no. 1, pp. 75–93, 1994.

[12] M. Viberg, "Subspace-based methods for the identification of linear time-invariant systems," *Automatica*, vol. 31, no. 12, pp. 1835–1851, 1995.

[13] L. Ljung, *System identification: Theory for the user*, 2nd ed., ser. Prentice Hall information and system sciences series. Upper Saddle River, NJ: Prentice Hall PTR, 2007.

[14] K. Kameyama, A. Ohsumi, U. Matsü, and K. Sawada, "Recursive 4sid based identification algorithm with fixed input output data size," *International journal of innovative computing, information & control: IJICIC*, vol. 1, 2005.

[15] H. Oku, "Recursive subspace model identification algorithms for slowly time-varying systems in closed loop," in *European Control Conference (ECC)*. IEEE, 2007, pp. 5715–5720.

[16] P. van Overschee and B. de Moor, "N4sid: Numerical algorithms for state space subspace system identification," *IFAC Proceedings Volumes*, vol. 26, no. 2, pp. 55–58, 1993.

[17] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[18] W. Xiong, L. Wu, F. Alleva, J. Droppo, X. Huang, and A. Stolcke, "The microsoft 2017 conversational speech recognition system," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 5934–5938.

[19] V. S. Murahari and T. Plötz, "On attention models for human activity recognition," in *ACM International Symposium on Wearable Computers - ISWC '18*, 2018, pp. 100–103.

[20] P. Wolf, A. Mrowca, T. T. Nguyen, B. Baker, and S. Gunnemann, "Pre-ignition detection using deep neural networks: A step towards data-driven automotive diagnostics," in *IEE 21st International Conference on Intelligent Transportation Systems (ITSC)*, 2018, pp. 176–183.

[21] Diederik P. Kingma and Jimmy Ba, "Adam: A method for stochastic optimization," in *3rd International Conference on Learning Representations (ICLR 2015).*, vol. abs/1412.6980, 2015.

4.2 Measurement Deviation Correction

Based on the paper printed in section 4.1, the following paper proposes an approach to correct detected measurement deviations with the help of Sparsity Constraints. We develop the classifier from section 4.1 further in such a way that it is able to distinguish between the absence and presence of hardware faults and measurement faults respectively. The core of the paper are the following two assumptions.

A1 For a sufficiently large N, and at each time step t, $\mathbf{i}_s(t)$ can be accurately approximated by the average of the simulated currents $\mathbf{i}^j_{sim}(t)$ across the N vehicles in the fleet, i.e. we have

$$\mathbf{i}_s(t) \approx \tilde{\mathbf{i}}_s(t) := \frac{1}{N} \sum_{j=1}^{N} \mathbf{i}^j_{sim}(t).$$

A2 The vector $\mathbf{x}$ containing all parameters $(d_h, \mathbf{o}_h, \mathbf{D}_m, \mathbf{o}_m)$ is a sparse vector, meaning that only few of the therein included parameters are non-zero.

Assumption A1 is the formal validation of Hypotheses H1 and H2b. Its justification is stated formally in the paper and detailed in this thesis in section 1.3.

Assumption A2 is the sparsity assumption. It assumes that it is very unlikely for an individual EV that all its HV current measurements of all sensors suffer from hardware and measurement faults at the same time. As mentioned in section 1.3, it is indeed very unlikely that hardware and measurement faults occur at newly produced EVs directly after the production. It is even unlikelier that this happens to all measurements at the same time. This would mean that all tests at the assembly line delivered false negatives. In the paper, we explain detailed how we use Quadratic Basis Pursuit (QBP) to recover the values vector $\mathbf{x}$. From $\mathbf{x}$, we retrieve the drift and offset values for hardware and measurement faults. Knowing these values allows us to send them back to each EV and to correct the measurements there. Thus, Hypothesis H3 is validated here.

Theoretically, the proposed approach works without parameters which would be required to be adjusted manually. In the paper, we explain that the new approach is even able to work without the distinction between hardware and measurement faults. However, from a practical perspective, it might be useful to work with such parameters to define a threshold under which no measurement correction is executed. This allows a more stable execution, as for example jumps between two or more fault parameters are suppressed. Thus, Hypothesis H4 is validated, because there is a solution without additional work for applicators. Nevertheless, it might be desirable to increase the workload for the applicators here in order to achieve a better system performance.

My contribution to this paper is, besides stating the fundamental problem and illustrating the corresponding Figures 1-4, the conceptualization for possible solutions. Ahmed Ayadi supports me here by considering the methodology of sparsity constraints and QBP, and by implementing the software to solve the problem. My responsibility

is the research into the state of the art, the collection and provision of data as well as the supervision and administration of the project. Furthermore, I am responsible for reading and editing the whole paper before the first submission as well as for editing the text to meet the reviewers' requirements during the review process.

IEEE Open Journal of
Intelligent Transportation
Systems

Received 29 January 2020; revised 18 May 2020; accepted 19 July 2020. Date of publication 22 July 2020; date of current version 12 August 2020.
The review of this article was arranged by Editor P. Khayyer.

Digital Object Identifier 10.1109/OJITS.2020.3011193

Measurement Correction for Electric Vehicles Based on Compressed Sensing

AHMED AYADI[1] AND JAKOB PFEIFFER[1,2]

[1]Department of Electrical and Computer Engineering, Technical University of Munich, 80333 Munich, Germany

[2]Function and Software Development E-Powertrain, BMW Group, 80788 Munich, Germany

CORRESPONDING AUTHOR: J. PFEIFFER (e-mail: jakob.j.pfeiffer@bmw.de)

This work was supported by the BMW Group.

ABSTRACT Deviations between system current measurements and real values in the power train of Electric Vehicles (EVs) can cause severe problems. Among others, these are restricted performance and cruising range. In this work, we propose a fleet-based framework to correct such deviations. We assume that the real value is the mean of all identically constructed EVs' measurements for the same input. Under this assumption, we decide for each vehicle whether it displays hardware errors with the help of a binary classifier. Depending on the classification, if no hardware errors are detected, we recover the parameters of an assumed measurement error model via Linear Regression. Otherwise, we combine the regression with a convex optimization problem and sparsity constraints. We achieve an overall recovery rate of up to 90%, allowing the full automation of the measurement correction procedure with no need to add more sensors, or computational units on-board of the EV.

INDEX TERMS Measurement correction, electric vehicles, machine learning, compressed sensing, sparsity constraints, intelligent sensors.

I. INTRODUCTION

MEASUREMENTS of High Voltage (HV) currents play a crucial role in the power train of EVs. For example, when the HV system operates close to its physical limits, measurements serve as base for the power limitation. Harming the limit would result in restricted battery life time and threaten safe vehicle operation [1]. The EV's safety is usually ensured with the help of battery protection offsets. The magnitude of the offsets depends on the measurement accuracy to make sure that it is sufficient even under the worst measurement conditions (compare Figure 1). Thus, inaccurate measurements lead to high offsets. High offsets are problematic for several reasons. One reason is that during acceleration the offset leads to a restriction of the power even if the measurement is more accurate than the worst expected and the power-train indeed would be able to provide the requested power. In this case, the EV's performance is restricted unnecessarily. An even worse problem occurs in the opposite situation. During recuperation a too conservatively chosen offset leads to a restriction of the amount of power which is charged into the High Voltage Battery (HVB) although the HVB would be capable to store it. This decreases the EV's cruising range.

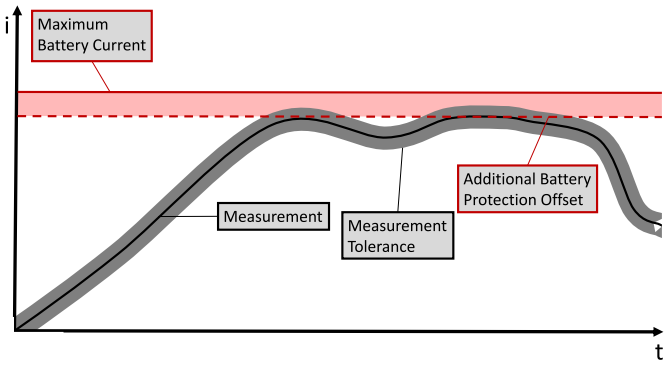


FIGURE 1. A schematic representation of power limitation due to battery protection during the acceleration of an EV. The measurement (black) might differ from the real value by some measurement tolerance (grey). To guarantee that the real value never exceeds the maximum battery current (red, solid), an additional offset (red, dashed) serves as prevention. The same principle is used analogously with negative currents during recuperation [2].

Kirchhoff's current law states that the sum of all currents at an electric node is equal to 0 A. As our HV system consists

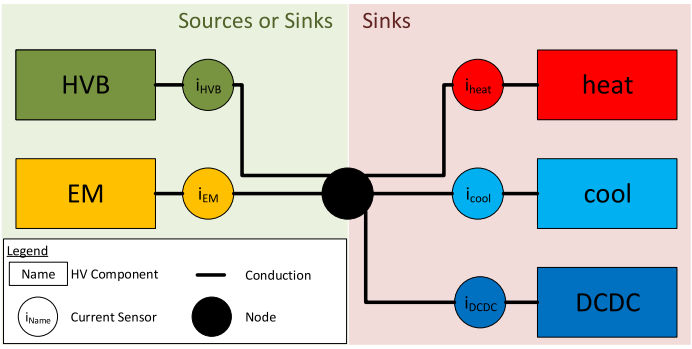


FIGURE 2. A highly simplified view of the HV power train of the EVs which we use for our studies. It consists of only a single node. Thus, the sum of all HV currents must be equal to 0 according to *Kirchhoff's current law*.

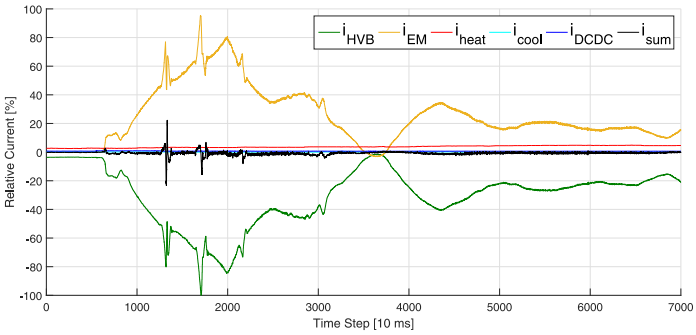


FIGURE 3. Currents of all HV components in an EV on a test drive. The sum of all currents i_{sum} is plotted in black. According to *Kirchhoff's current law*, it should be constantly 0 %. But the measurements show that the deviation i_{sum} is higher than the current of the DCDC converter i_{DCDC}.

of only a single node (compare Figure 2), the sum of all HV currents must be equal to 0 A according to Kirchhoff's law. However, considering measurement signals of EVs with distributed sensor systems, the sum of all currents can differ by up to 25 % of the maximum current (see Figure 3). If we look closer at the Root Mean Square Error (RMSE) of the sum of currents $\text{RMSE}(i_{\text{sum}}) = 0.67\%$, we realize that it has on average the same value as the current of the DCDC converter which is $\mu_{i_{\text{DCDC}}} = 0.67\%$.

Our aim is to increase the measurement accuracy to enable a minimization of the battery protection offsets. To this end, we develop and evaluate a measurement correction system based on the measurement fault detection we propose in [3]. The correction uses compressed sensing and sparsity constraints. It works based on the power train data alone and does not require further expert knowledge for manual calibration.

We explain related work and our contribution to the state of the art in Section II. In Section III, we explain the theory behind our work before we describe the practical experiments in Section IV. The results of our experiments are stated in Section V. In Section VI, we discuss the advantages and drawbacks of the proposed concepts. Finally, we draw our conclusions in Section VII.

II. STATE OF THE ART

In this work, our contribution is the development of an automated measurement correction system based on compressed sensing. We enhance the measurement deviation detection presented in [3] in such a way that it is not only able to detect, but also to recursively correct measurement faults. The proposed system is able to minimize deviations between measurements and real values. The self-reliant correction uses methods from the field of Machine Learning and does not require any expert knowledge or high calibration effort for its execution. We develop the measurement system close to its field of application. Thus, it works with only the data already available in modern series EVs without additional sensors. Another advantage of our system is that it does not increase the computation load or memory consumption of the Electronic Control Units (ECUs) of EVs. We evaluate our approach with simulation data based on previously recorded real power train data of series EVs on public roads. To the best of the authors' knowledge, this is the first time that a measurement correction system is proposed for the HV power trains of close-to-production EVs based on compressed sensing without redundant sensor systems.

Within the scope of our work, we differentiate between two different kinds of faults: *measurement* and *hardware faults*. Hardware faults describe sensors measuring correctly wrong behavior, e.g., in the case of broken actuators. Speaking of measurement faults, we mean faulty measurement data caused by, e.g., corrupted sensors. Besides the two kinds of faults, we distinguish between two groups of Machine Learning approaches for measurement correction: *off-board* and *on-board* approaches.

Off-board approaches train a model outside of its common environment. The training is based on previously recorded data or simulation. After training, the model is executed on-board and detects deviations from the previously learned behavior.

Malakar *et al.* [4] increase the quality of their measurements with an off-board approach based on Neural Networks. They detect and neglect input signals which lead to a bias in the measurement output to improve the measurement quality. However, even corrupted signals might contain parts with valuable information [5]. Neglecting the whole signal means to drop also the valid parts of the information. Thus, we prefer to correct corrupted signals instead of dropping them. Malakar's measurement environment consists only of the sun and the air. There are no further actuators. In contrast to our work, deviations in their data cannot be caused by hardware faults and are always provoked by measurement faults.

Hardware and measurement faults are distinguished by Zhao *et al.* [6]. The authors construct a simulation model of an aero-engine. Their model contains sub-models based on physical principles. The sub-models represent all components including actuators and sensors. Zhao *et al.* detect deviations between expected and measured values with the help of

Principal Component Analysis (PCA) and diagnosis models. They are able to recognize sensor faults and the affected system components due to the component-wise modeling. This modeling technique demands expert knowledge about the physical principles influencing the measurement environment. If the physical modeling is not carried out accurately enough, it can lead to a problem which is known as the *reality gap* in evolutionary robotics [7]. The reality gap denotes the phenomenon that models perform well during simulation, but fail once they are executed in the real world [5]. A reason for this failure is that the simulation data used for training is often only available in certain working conditions, whereas the environment of the real system varies across a broad range during execution [8].

The reality gap is a general problem of off-board approaches. It is their main drawback that situations might occur during execution which the algorithm did not experience during training. As a result, these algorithms are not able to adapt to new circumstances and thus perform suboptimally in certain situations [8]. Another drawback is that the training of Machine Learning methods usually is quite time-consuming [5]. Off-board approaches separate this expensive training from low-performance execution platforms. Thus, they allow cheaper hardware which is an advantage in cost-efficient industries like the automotive domain. Another advantage is that the separated training environment allows the use of higher computation and memory resources. This enables a broad range of algorithms to be considered for solving the requested problem. Additionally, the training is not necessarily required to be executed in real-time.

The major difference of on-board approaches is that the models are directly trained on the execution platform and then updated continuously.

An example for an on-board approach is the DC current calibration presented by Ren *et al.* with an high-precision current adder [9]. The adder is an additional hardware component which is able to correct faulty measurements during execution. The authors apply the adder successfully in the electrolysis industry. However, the automotive industry has different requirements. Due to the restricted available installation space and the cost efficiency resulting from mass production, we want to avoid additional hardware. Our algorithms are supposed to run with the hardware and the measurement data available in modern series vehicles.

The sensor set of a series production engine is sufficient for Lu *et al.*'s inspiring approach [8]. They introduce an on-board approach for sensor fault detection with an Extreme Learning Machine and apply it to the control system of an aero-engine. Although their approach is capable to detect bias and drift faults, it is not able to distinguish between hardware and measurement faults. Thus, if a hardware fault occurs, it must be detected separately. Nevertheless, their approach can correct measurement faults by providing compensation data.

Like Lu *et al.*, Kobayashi and Simon focus on the detection of faults in an aero-engine [10]. They propose an on-board approach which is capable to differentiate between hardware and measurement faults. Instead of an Extreme Learning Machine, Kobayashi and Simon use a bank of Kalman Filters. Each filter monitors a sensor signal separately. Kobayashi and Simon create an additional signal to detect hardware faults. They isolate corrupted signals with the help of a decision matrix. However, particularly for problems with many signals, a filter for each signal leads to a high number of filters and thus to high computational costs.

On the one hand, on-board approaches have the advantage of continuously updating their model. Thus, these algorithms are capable to adapt to never before experienced situations. On the other hand, they suffer from three main disadvantages. First, the training is required to be executed in real-time for many use cases. Second, cost effective design prevents to add additional performance and memory resources to ECUs. Especially in the automotive domain, this is an issue of high interest and restricts the capability of learning on-board of EVs. As a result, many algorithms become infeasible for automotive ECUs. Third, on-board approaches are only able to detect deviating behavior of otherwise working sensors. If the sensor returns biased measurements from the initial execution, the data is mistakenly assumed to be correct.

In our previous work, we develop a fleet-based approach with an on-board trained measurement model and an off-board fault classifier for close-to-production EVs [3]. With that hybrid approach, we combine the ability to handle unseen situations with detecting ab initio corrupted sensors. The on-board measurement model minimizes the data transferred over the air. This enables us to use the resources of a back end. In the back end, the classifier is capable to differentiate between hardware and two kinds of measurement faults. The measurement deviation detection proposed there serves as basis for our work in this paper. Here, we develop the classifier further. Additionally, we append the still missing correction of deviations between measurements and real values.

In their inspiring paper, Ohlsson *et al.* extend classical compressive sensing to quadratic relations and second order Taylor expansions [11]. They give examples for different types of measurements. Their paper serves as theoretical basis for our work described in this article. We want to extend their approach to our measurements to correct measurement faults.

Besides measurement faults and sensor uncertainties, the divergence between measurements and real values can be caused by time delays [12]. Time delays are not the focus of this work. For this work, we assume that all data is synchronized correctly and potential time delays have been detected and corrected previously. We treat with the detection of time delays in our other previous work [2], [12].

III. CONCEPTS

In the HV power train of an EV, we consider an electric system of K currents $i_1, i_2, \ldots, i_K$. To each current i_k, we dedicate one sensor providing measurements of i_k

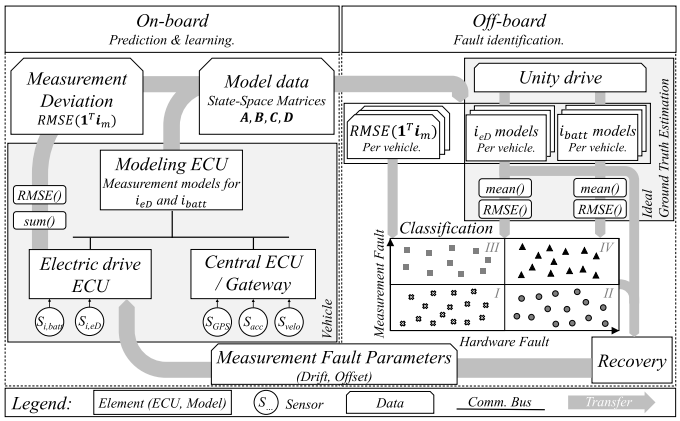


FIGURE 4. A graphical illustration of our approach. For the sake of simplicity, our approach is illustrated here only for the electric machine and the HVB. The reduction to two HV component means that the current vectors $\mathbf{i} = (i_{\text{eD}}, i_{\text{batt}})$ and $\mathbf{i}_m = (i_{\text{eD},m}, i_{\text{batt},m})$ are of dimension $K = 2$ in this case.

in real-time (without time delays), which we henceforth denote by $i_{k,m}$. We further assume that all K currents flow into the same node, so that we have their sum $\sum_{k=1}^{K} i_k$ equal to 0 by *Kirchhoff's current law* (because of measurement errors, this needs not to be true for the sum $\sum_{k=1}^{K} i_{k,m}$). To simplify notation, we denote from now on by $\mathbf{i}$ and $\mathbf{i}_m$ the K-dimensional vectors $(i_1, i_2, \ldots, i_K)^T$ and $(i_{1,m}, i_{2,m}, \ldots, i_{K,m})^T$, respectively.

Our approach is illustrated in Figure 4. We presume an EV collecting measurement data $\{\mathbf{i}_m(t)\}_{1\le t\le T}$ during driving. Based on these data, we state our problem as that of learning a correction mapping $\mathbf{i}_m \mapsto \mathbf{i}$. Our aim is to subsequently use this mapping to correct the measurement signals in real-time. The main challenge thereby is that only $\mathbf{i}_m(t)$ is available, but not the corresponding ground truth $\mathbf{i}(t)$. The lack of the ground truth makes it impossible to learn the desired mapping a priori in a supervised manner.

Thus, we want to estimate an *ideal* ground truth $\mathbf{i}_s$ first (see Section III-A). Then, a pre-trained classifier decides whether the considered EV displays hardware faults (see Section III-B). Finally, depending on the classification result, a recovery algorithm is executed to learn the desired mapping (see Section III-C).

A. IDEAL GROUND TRUTH ESTIMATION

In the sequel, we suppose the existence of a *perfect* vehicle from the investigated model, i.e., a vehicle S featuring a perfect behavior with respect to the model specifications, and equipped with perfect sensors. Let $i_{k,s}$ be the k-th current in S, and $\mathbf{i}_s = (i_{1,s}, i_{2,s}, \ldots, i_{K,s})^T$. Then, we can think of $\mathbf{i}_s$ as the should-be value of $\mathbf{i}_m$ resulting in $\mathbf{i}_s = \mathbf{i}_m$. Thus, for any *imperfect* vehicle we retrieve $\mathbf{i}_m \neq \mathbf{i}_s$, whereby this deviation can be caused by hardware faults, measurement faults or a combination of both.

On the one hand, because of potential hardware faults (e.g., a flat tire), the vehicle might display a dynamical behavior different from that of S, which then changes $\mathbf{i}_s$ into $\mathbf{i}$, but without changing the sum of currents which remains 0, i.e., $\mathbf{1}^T\mathbf{i} = \mathbf{1}^T\mathbf{i}_s = \mathbf{0}$.

On the other hand, because of measurement errors (e.g., a sensor offset/drift), a second deviation might be observed between $\mathbf{i}_m$ and $\mathbf{i}$. In contrast to hardware errors, this deviation does most likely change the sum of currents, so that in general one has $\mathbf{1}^T\mathbf{i}_m \neq \mathbf{0}$. Together, we can write

$$\underbrace{\mathbf{i}_m - \mathbf{i}_s}_{\text{total deviation}} = \underbrace{\mathbf{i}_m - \mathbf{i}}_{\text{measurement error}} + \underbrace{\mathbf{i} - \mathbf{i}_s}_{\text{hardware error}} . \tag{1}$$

We assume the existence of a fleet of N (imperfect) vehicles from the investigated model, and denote the vector of measured currents in the j-th vehicle by $\mathbf{i}_m^j$. For each j, we train a discrete *State-Space Model* consisting of the system matrices $\mathbf{A}^j$, $\mathbf{B}^j$, $\mathbf{C}^j$ and $\mathbf{D}^j$ with state $\mathbf{x}$, input $\mathbf{u}$ and output $\mathbf{i}_m^j$ based on a set $\{\mathbf{i}_m^j(t)\}_{1\le t\le T^j}$ of measurement data as described in [3]. In the next step, we estimate the currents $\{\mathbf{i}_m^j\}_{1\le j\le N}$ that would be measured across the fleet, if all vehicles drove under the exactly same conditions [3]. Simulated on these *unity drive* conditions, $\{\mathbf{i}_{sim}^j(t)\}_{1\le j\le N}$ denotes the outputs of the trained measurement models at time t according to

$$\begin{aligned} \mathbf{x}(t+1) &= \mathbf{A}^j\mathbf{x}(t) + \mathbf{B}^j\mathbf{u}_{ud}(t) \\ \mathbf{i}_{sim}^j(t) &= \mathbf{C}^j\mathbf{x}(t) + \mathbf{D}^j\mathbf{u}_{ud}(t) \end{aligned} \tag{2}$$

with $\mathbf{x}_0 = \mathbf{x}(0)$ and $\mathbf{u}(t) = \mathbf{u}_{ud}(t)$, $t \in \{1, 2, \ldots, T_{ud}\}$.

Finally, by denoting the current vector $\mathbf{i}_s(t)$ of the perfect vehicle S at time t under the unity drive conditions, we come to the following assumption:

Assumption 1: For a sufficiently large N, and at each time step t, $\mathbf{i}_s(t)$ can be accurately approximated by the average of the simulated currents $\mathbf{i}_{sim}^j(t)$ across the N vehicles in the fleet, i.e., we have

$$\mathbf{i}_s(t) \approx \tilde{\mathbf{i}}_s(t) := \frac{1}{N}\sum_{j=1}^{N} \mathbf{i}_{sim}^j(t). \tag{3}$$

This assumption is based on the observation

$$\begin{aligned} \frac{1}{N}\sum_{j=1}^{N} \mathbf{i}_{sim}^j &= \frac{1}{N}\sum_{j=1}^{N}\Big(\mathbf{i}_s + \mathbf{i}_m^j - \mathbf{i}^j + \mathbf{i}^j - \mathbf{i}_s + \mathbf{i}_{sim}^j - \mathbf{i}_m^j\Big) \\ &= i_s + \frac{1}{N}\sum_{j=1}^{N}\Big(\mathbf{i}_m^j - \mathbf{i}^j\Big) + \frac{1}{N}\sum_{j=1}^{N}\Big(\mathbf{i}^j - \mathbf{i}_s\Big) \\ &\quad + \frac{1}{N}\sum_{j=1}^{N}\Big(\mathbf{i}_{sim}^j - \mathbf{i}_m^j\Big) \end{aligned} \tag{4}$$

and the assumption that for large N, measurement, hardware and simulation errors at a certain time step average to 0 across the fleet, so that the three averages in (4) converge to 0.

B. CLASSIFICATION

Having obtained an estimation for $\mathbf{i}_m^j$ and $\mathbf{i}_s$ at each time step $t \in \{1, 2, \ldots, T_{ud}\}$, namely $\mathbf{i}_m^j(t) \approx \mathbf{i}_{sim}^j(t)$ and $\mathbf{i}_s(t) \approx$

$\tilde{\mathbf{i}}_s(t)$ as in (3), we would have almost solved the problem for the j-th vehicle, if we could exclude the possibility of hardware errors. In fact, if that was the case, we would have $\mathbf{i}^j(t) = \mathbf{i}_s(t) \approx \tilde{\mathbf{i}}_s(t)$, which we could use together with $\mathbf{i}^j_{sim}(t)$ to learn the desired mapping $\mathbf{i}^j_m \mapsto \mathbf{i}^j$ (i.e., the mapping is learned by means of the tuples $\{(\mathbf{i}^j_{sim}(t), \tilde{\mathbf{i}}_s(t))\}_{1\le t\le T_{ud}}$).

In order to reduce the complexity of our approach, we propose to first predict by means of a classifier whether any given EV displays hardware errors. If the prediction is negative (i.e., no hardware errors are detected), we proceed as described above. Otherwise, we utilize a more complex algorithm to recover the measurement error (see Section III-C). In this work, we propose and compare two different classification rules, both based on a set of $K+1$ features $f_1, f_2, \ldots, f_{K+1}$.

On the one hand, it seems reasonable that the absolute total deviation between $\mathbf{i}^j_m$ and $\mathbf{i}_s$ would in average be larger, if the j-th EV suffers not only from measurement faults but also from significant hardware faults (see (1)). Accordingly, we estimate by means of $\mathbf{i}^j_{sim}(t)$ and $\tilde{\mathbf{i}}_s(t)$ for each vehicle j and each current k the average quadratic deviation between $i^j_{k,m}$ and $i_{k,s}$, and define that as the k-th feature for the j-th vehicle

$$f^j_k = \frac{1}{T_{ud}} \sum_{t=1}^{T_{ud}} \left(i^j_{k,sim}(t) - \tilde{i}_{k,s}(t) \right)^2 \approx \mathbb{E}\left(\left(i^j_{k,m} - i_{k,s} \right)^2 \right) \quad (5)$$

for $1 \le k \le K$.

On the other hand, we define the $(K+1)$-th feature as a measure of the significance of measurement faults in the j-th vehicle. Since these faults result in $\mathbf{i}^j_m$ almost surely not summing to 0, we quantify the magnitude of the measurement faults by means of the average square of $\mathbf{1}^T\mathbf{i}^j_m$, i.e.,

$$f^j_{K+1} = \frac{1}{T^j} \sum_{t=1}^{T^j} \left(\left(\mathbf{1}^T \mathbf{i}^j_m(t) \right)^2 \right) \approx \mathbb{E}\left(\left(\mathbf{1}^T \mathbf{i}^j_m \right)^2 \right), \quad (6)$$

where a set $\{\mathbf{i}^j_m(t)\}_{1\le t\le T^j}$ of measurement data is required for the estimation.

Let $\mathbf{f}^j = (f^j_1, f^j_2, \ldots, f^j_{K+1})^T$. Based on $\mathbf{f}^j$, we decide whether the j-th vehicle displays significant hardware faults. We thereby compare between two classification rules:

- *Simple Thresholding Classifier:* The classifier decides **for** the existence of hardware errors, if

$$\eta^j := \sum_{i=1}^{K} f^j_i - \frac{1}{K} f^j_{K+1} > \delta_h \quad (7)$$

for some threshold δ_h. For an intuitive explanation, we rewrite the above criterion as

$$\eta^j \approx \sum_{k=1}^{K} \mathbb{E}\left(\left(i^j_{k,m} - i_{k,s} \right)^2 \right) - \frac{1}{K} \mathbb{E}\left(\left(\mathbf{1}^T \mathbf{i}^j_m \right)^2 \right)$$

$$= \mathbb{E}\left(\sum_{k=1}^{K} \left(i^j_{k,m} - i_{k,s} \right)^2 - \left(\frac{\mathbf{1}^T \mathbf{i}^j_m}{\sqrt{K}} \right)^2 \right)$$

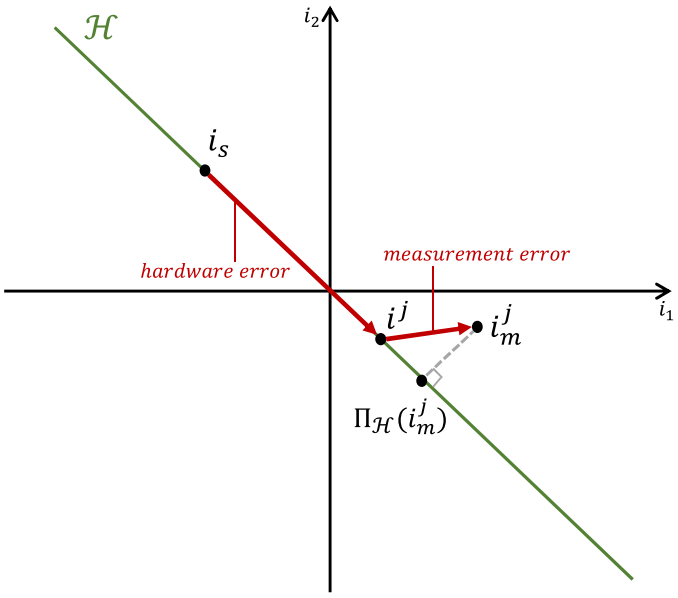


FIGURE 5. Graphical representation for $K = 2$. Here, $\mathcal{H}$ is the line described by $i_1 + i_2 = 0$. $\mathbf{i}_s$ is the should-be current and part of $\mathcal{H}$ as it satisfies *Kirchhoff's law*. Hardware errors in the j-th vehicle move $\mathbf{i}_s$ to another point on $\mathcal{H}$, here $\mathbf{i}^j$. Then, measurement errors move $\mathbf{i}^j$ outside of $\mathcal{H}$ to $\mathbf{i}^j_m$. When projected back on $\mathcal{H}$, $\mathbf{i}^j_m$ returns $\Pi_{\mathcal{H}}(\mathbf{i}^j_m)$. In (8), we use *Pythagoras theorem* on the triangle spanned by the points $\mathbf{i}_s$, $\mathbf{i}^j_m$ and $\Pi_{\mathcal{H}}(\mathbf{i}^j_m)$. The triangle is right-angled at $\Pi_{\mathcal{H}}(\mathbf{i}^j_m)$.

$$= \mathbb{E}\left(\left\| \mathbf{i}^j_m - \mathbf{i}_s \right\|_2^2 - \left\| \mathbf{i}^j_m - \Pi_{\mathcal{H}}\left(\mathbf{i}^j_m \right) \right\|_2^2 \right) > \delta_h \quad (8)$$

by inserting (5). Here, $\Pi_{\mathcal{H}}$ is the projection operator on the hyperplane $\mathcal{H}$ given by $\mathcal{H} = \{\mathbf{x} \in \mathbb{R}^K \mid \mathbf{1}^T\mathbf{x} = 0\}$. Since $\mathbf{i}_s$ satisfies *Kirchhoff's law*, we have $\mathbf{i}_s \in \mathcal{H}$, and thus by *Pythagoras* (see Figure 5)

$$\left\| \mathbf{i}^j_m - \mathbf{i}_s \right\|_2^2 - \left\| \mathbf{i}^j_m - \Pi_{\mathcal{H}}\left(\mathbf{i}^j_m \right) \right\|_2^2 = \left\| \Pi_{\mathcal{H}}\left(\mathbf{i}^j_m \right) - \mathbf{i}_s \right\|_2^2. \quad (9)$$

Knowing that $\mathbf{i}^j$ also satisfies *Kirchhoff's law* and thus $\mathbf{i}^j \in \mathcal{H}$, we can think of $\Pi_{\mathcal{H}}(\mathbf{i}^j_m)$ as the best possible approximation of $\mathbf{i}^j$ in the absence of any further information. Thus, the above criterion is an approximation of the average squared distance $||\mathbf{i}^j - \mathbf{i}_s||_2^2$, i.e., of the average squared deviation caused by hardware faults (see (1)).

- *Decision Tree Classifier:* The classifier decides whether or not the j-th vehicle displays hardware faults based on previously induced comparison rules. The rules are learned during a training phase with an artificially created fleet of EVs. The artificial creation is based on data from real drives and has the advantage for our work that we know about the (non)-existence of hardware faults in advance. Thus, we are able to evaluate the correction.

Based on the utilized classifier, we distinguish between four different cases:

- *Case 1:* Neither hardware nor measurement faults are detected. In this case, we have a nearly *perfect* vehicle, for which no further steps are necessary.

- *Case 2:* Only hardware faults are detected. No further steps are necessary in this case, too, as our goal is to correct measurement faults only.
- *Case 3:* Only measurement faults are detected. Here, only one further step is necessary, namely to use the tuples $\{(\mathbf{i}^j_{sim}(t), \mathbf{i}_s(t))\}_t$ to learn the correction mapping $\mathbf{i}^j_m \mapsto \mathbf{i}^j$.
- *Case 4:* Both types of faults are detected, i.e., the deviation between $\mathbf{i}^j_m$ and $\mathbf{i}_s$ is to be decomposed into its two components as in (1). Without further assumptions, the problem is however ill-posed, as there is an infinite number of such decompositions. In order to provide a unique solution, we propose in the sequel a set of assumptions which we find to be both reasonable and sufficient to make the problem well-posed again.

C. ERROR MODEL AND RECOVERY

In this section, we first present the error models we assume for measurement and hardware faults. Since hardware faults do not alter the validity of physical laws, here in particular *Kirchhoff's law*, one can in general model them using a continuous mapping $\mathbf{f}_h : \mathbf{i}_s \mapsto \mathbf{i}$, such that it holds

$$\mathbf{i}_s \in \mathcal{H} \implies \mathbf{f}_h(\mathbf{i}_s) \in \mathcal{H}. \tag{10}$$

In this work, we choose a simple linear model for $\mathbf{f}_h$ satisfying (10), namely:

$$\mathbf{f}_h(\mathbf{i}_s) = (1 + d_h)\mathbf{i}_s + \mathbf{o}_h, \tag{11}$$

where $d_h \in \mathbb{R}$ is a drift scalar and $\mathbf{o}_h \in \mathcal{H}$ is an offset vector. We retrieve

$$\begin{aligned} \mathbf{i}_s, \mathbf{o}_h \quad &\in \quad \mathcal{H} \\ &\implies \mathbf{1}^T\mathbf{i}_s = \mathbf{1}^T\mathbf{o}_h = \mathbf{0} \\ &\implies (1 + d_h)\mathbf{1}^T\mathbf{i}_s + \mathbf{1}^T\mathbf{o}_h = \mathbf{0} \\ &\implies \mathbf{1}^T\mathbf{f}_h(\mathbf{i}_s) = \mathbf{0} \\ &\implies \mathbf{f}_h(\mathbf{i}_s) \in \mathcal{H}. \end{aligned} \tag{12}$$

Similarly, we model measurement faults as a linear mapping $\mathbf{f}_m : \mathbf{i} \mapsto \mathbf{i}_m$. However, we model the mapping without the constraints from (10), since the noisy vector $\mathbf{i}_m$ does not need to satisfy *Kirchhoff's law*. This results in

$$\mathbf{f}_m(\mathbf{i}) = (\mathbf{I} + \mathbf{D}_m)\mathbf{i} + \mathbf{o}_m, \tag{13}$$

where $\mathbf{I}$ is the identity matrix, $\mathbf{D}_m \in \mathbb{R}^{K\times K}$ is a diagonal drift matrix, and $\mathbf{o}_m \in \mathbb{R}^K$ is an offset vector. Note that in these terms, the goal of this work is to learn the inverse mapping $\mathbf{f}_m^{-1}(\mathbf{i}_m)$. As $\mathbf{i}$ is unknown, we instead aim to learn the parameters $\mathbf{D}_m$ and $\mathbf{o}_m$ with the help of the tuples $(\mathbf{i}_m, \mathbf{i}_s)$.

From (11) and (13), we retrieve

$$\begin{aligned} \mathbf{i}_m &= \mathbf{f}_m \circ \mathbf{f}_h(\mathbf{i}_s) = (\mathbf{I} + \mathbf{D}_m)((1 + d_h)\mathbf{i}_s + \mathbf{o}_h) + \mathbf{o}_m \\ &= (\mathbf{I} + \mathbf{D}_m)(1 + d_h)\mathbf{i}_s + (\mathbf{I} + \mathbf{D}_m)\mathbf{o}_h + \mathbf{o}_m. \end{aligned} \tag{14}$$

However, we cannot generally recover the parameters (in particular $\mathbf{D}_m$ and $\mathbf{o}_m$) from tuples in the form $(\mathbf{i}_m, \mathbf{i}_s)$ (and even less $(\mathbf{i}_{sim}, \tilde{\mathbf{i}}_s)$). To solve this problem, we consider the classification result and differentiate between two cases.

1) RECOVERY WITH NO HARDWARE ERRORS

This case is equivalent to setting both d_h and $\mathbf{o}_h$ to 0. We retrieve $\mathbf{i}_m = (\mathbf{I}+\mathbf{D}_m)\mathbf{i}_s+\mathbf{o}_m$ from (14). We learn the parameters $\mathbf{D}_m$ and $\mathbf{o}_m$ straightforward by linearly regressing $\mathbf{i}_{sim}(t)$ on $\tilde{\mathbf{i}}_s(t)$. More precisely, we execute in total K regressions where we regress $i_{sim,k}(t)$ on $i_{m,k}(t)$ for each k.

2) RECOVERY WITH HARDWARE ERRORS

Here, the problem is ill-posed without additional knowledge. For example, if $(d_h, \mathbf{o}_h, \mathbf{D}_m, \mathbf{o}_m)$ satisfies (14), then $(d_h, \mathbf{o}_h + \mathbf{e}, \mathbf{D}_m, \mathbf{o}_m - (\mathbf{I} + \mathbf{D}_m)\mathbf{e})$ does so, too, for any $\mathbf{e} \in \mathcal{H}$. In this work, we avoid such ill-posedness with the following assumption.

Assumption 2: The vector $\mathbf{x}$ containing all parameters $(d_h, \mathbf{o}_h, \mathbf{D}_m, \mathbf{o}_m)$ is a sparse vector, meaning that only few of the therein included parameters are non-zero.

This assumption bases upon the following reasoning: in one vehicle, it is very unlikely that all, or at least many, fault sources exist concurrently. All faults at the same time would mean that all K involved sensors suffer from offset or drift faults, or even both. Additionally, K currents are affected by hardware faults without exception. Thus, it is reasonable to assume that only relatively few sources of faults exist in the same vehicle at the same time.

Accordingly, we recover $\mathbf{x}$ as follows. First, we estimate the expressions $(\mathbf{I}+\mathbf{D}_m)(1+d_h)$ and $(\mathbf{I}+\mathbf{D}_m)\mathbf{o}_h+\mathbf{o}_m$ based on the tuples $(\mathbf{i}_{sim}(t), \tilde{\mathbf{i}}_s(t))$. We do this estimation by linearly regressing $\mathbf{i}_{sim}(t)$ on $\tilde{\mathbf{i}}_s(t)$. More precisely, we define the k-th diagonal element of $\mathbf{D}_m$ as $d_{m,k}$, and the k-th elements of $\mathbf{o}_h$ and $\mathbf{o}_m$ as $o_{h,k}$ and $o_{m,k}$, respectively. Then, we retrieve for each $k \in \{1, 2, \ldots, K\}$

$$\begin{aligned} \big(1 + d_{m,k}\big)(1 + d_h) &\approx a_k \\ \big(1 + d_{m,k}\big)o_{h,k} + o_{m,k} &\approx b_k \end{aligned} \tag{15}$$

where a_k and b_k are the slope and intercept estimates obtained from regressing $(i_{sim,k}(1), \ldots, i_{sim,k}(T_{ud}))^T$ on $(\tilde{i}_{s,k}(1), \ldots, \tilde{i}_{s,k}(T_{ud}))^T$. Next, we define $\mathbf{y} = (a_1, \ldots, a_K, b_1, \ldots, b_K)^T$ and $\tilde{\mathbf{x}} = (d_h, o_{h,1}, \ldots, o_{h,K}, d_{m,1}, \ldots, d_{m,K}, o_{m,1}, \ldots, o_{m,K}, 1)^T \in \mathbb{R}^{3K+2}$. We then rewrite each of the estimated terms in a quadratic form $\tilde{\mathbf{x}}^T\mathbf{Q}\tilde{\mathbf{x}}$ of $\tilde{\mathbf{x}}$, for example

$$\begin{aligned} \big(1 + d_{m,k}\big)(1 + d_h) &= \Big(\tilde{\mathbf{x}}^T\mathbf{e}_{3K+2}\Big)^2 + \tilde{\mathbf{x}}^T\mathbf{e}_{3K+2}\mathbf{e}^T_{K+1+k}\tilde{\mathbf{x}} \\ &\quad + \tilde{\mathbf{x}}^T\mathbf{e}_{3K+2}\mathbf{e}^T_1\tilde{\mathbf{x}} + \tilde{\mathbf{x}}^T\mathbf{e}_{K+1+k}\mathbf{e}^T_1\tilde{\mathbf{x}} \\ &= \tilde{\mathbf{x}}^T\Big(\mathbf{e}_{3K+2}\mathbf{e}^T_{3K+2} + \mathbf{e}_{3K+2}\mathbf{e}^T_{K+1+k} \\ &\quad + \mathbf{e}_{3K+2}\mathbf{e}^T_1 + \mathbf{e}_{K+1+k}\mathbf{e}^T_1\Big)\tilde{\mathbf{x}} \\ &\coloneqq \tilde{\mathbf{x}}^T\mathbf{Q}_k\tilde{\mathbf{x}}, \end{aligned} \tag{16}$$

where the notation $\mathbf{e}_{3K+2}$ denotes the canonical vector with respect to $3K + 2$. Similarly, we obtain $(1+d_{m,k})o_{h,k}+o_{m,k} = \tilde{\mathbf{x}}^T\mathbf{Q}_{K+k}\tilde{\mathbf{x}}$ and rewrite (15) as

$$\forall i \in \{1, 2, \ldots, 2K\} \quad \tilde{\mathbf{x}}^T\mathbf{Q}_i\tilde{\mathbf{x}} \approx \mathbf{y}^T\mathbf{e}_i. \tag{17}$$

Finally, we recover $\mathbf{x}$ (resp. $\tilde{\mathbf{x}}$) by solving the optimization problem

$$\begin{aligned}\min_{\tilde{\mathbf{x}}} \ \|\tilde{\mathbf{x}}\|_1 \ \text{ s.t. } \ &\sum_{i=1}^{2K}\left(\tilde{\mathbf{x}}^T\mathbf{Q}_i\tilde{\mathbf{x}} - \mathbf{y}^T\mathbf{e}_i\right)^2 \leq \epsilon,\\ &\tilde{\mathbf{x}}^T\mathbf{e}_{3K+2} = 1,\\ &\tilde{\mathbf{x}}^T\left(\sum_{i=1}^{K}\mathbf{e}_{1+i}\right) = 0. \end{aligned} \tag{18}$$

Thereby, ϵ is an error threshold. The constraint $\tilde{\mathbf{x}}^T\mathbf{e}_{3K+2} = 1$ forces the last element of $\tilde{\mathbf{x}}$ to be equal to 1, and the constraint $\tilde{\mathbf{x}}^T(\sum_{i=1}^{K}\mathbf{e}_{1+i}) = 0$ makes sure that the hardware offsets $o_{h_1}, \ldots, o_{h,K}$ sum up to 0 to satisfy (10). To solve the optimization problem from (18), we use the Quadratic Basis Pursuit (QBP) algorithm suggested in [11]. This algorithm uses a lifting technique to convexify the above problem and thus makes it computationally tractable.

In the end of this section, we want to emphasize that in theory, we could bypass the classification step from Section III-B. Our proposed recovery does not require the existence of hardware errors. It could theoretically recover the parameters successfully even when $d_h = 0$ and $\mathbf{o}_h = \mathbf{0}$. Thus, we could directly execute the optimization in (18) for all EVs, irrespective of whether or not they are affected by hardware errors. We would only need to run the optimization procedure in (18) when necessary. This would reduce the complexity without negatively affecting the performance. However, in the presence of classification errors, the recovery' success rate might be negatively affected. On the one hand, in case of false positives, the algorithm would try to recover the parameters using the optimization problem from (18). Instead of the simple and highly reliable regression from Section III-C1, the optimization problem might fail to recover the parameters. On the other hand, in case of false negatives, the recovery algorithm would assume wrongly that there are no hardware errors. The wrong assumption would lead to a systematic estimation error. From these observations, we conclude that false negatives are more harmful than false positives in our case. We will take this fact into account for the evaluation of the classification step in Section V-B.

IV. EXPERIMENTAL SETUP

In this section, we describe the experiments we conduct in order to evaluate the suggested method.

A. ARTIFICIAL GROUND TRUTH

Each experiment is based on an artificially created data set $\mathcal{D} = \{\mathbf{i}_s(t)\}_{1\leq t\leq T}$ of ground truth current signals. To make sure that the data set is realistic, we start with real measurement data $\mathcal{D}' = \{\mathbf{i}_m(t)\}_{1\leq t\leq T}$ of HV current signals recorded during driving. The data set is modified in such a way that at each time step t, the condition $\mathbf{i}_s(t)^T\mathbf{1} = \mathbf{0}$ is satisfied. We guarantee the satisfaction of the condition by setting

$$i_{k,s}(t) = \begin{cases} i_{k,m}(t) & \text{for } k \in \{1, \ldots, K-1\} \\ -\sum_{j=1}^{K-1} i_{j,m}(t) & \text{for } k = K \end{cases}. \tag{19}$$

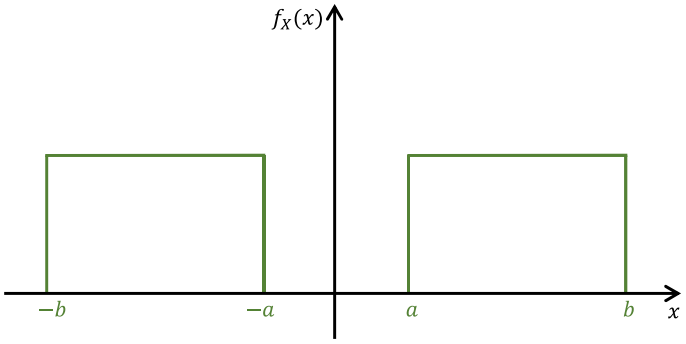


FIGURE 6. The density function.

B. ARTIFICIAL RANDOM FLEET

Besides the artificial ground truth, we create a random fleet of N EVs based on $\mathcal{D}$ and the model given in (14). We do this by randomly drawing the parameters $(d_h, \mathbf{o}_h, \mathbf{D}_m, \mathbf{o}_m)$ for each EV, so that

- the constraint $\mathbf{o}_h^T\mathbf{1} = \mathbf{0}$ is satisfied,[1]
- for a given sparsity level S, only $(100 - S)\%$ of the parameters (up to rounding) are nonzero. (For example, if $K = 3$ and $S = 60$, we have in total $3K + 1 = 10$ parameters. Only 4 of these are set to be nonzero. The support of the parameter vector is thereby chosen at random.),
- each nonzero parameter is set as the realization of some random variable $X = (2B - 1)U$ where

$$B \sim Bernoulli\left(\frac{1}{2}\right) \text{ and } U \sim \mathcal{U}(a, b) \tag{20}$$

for some hyper-parameters a and b. The corresponding density function is depicted in Fig. 6. The rationale behind this choice of the distribution is to prevent a previously nonzero parameter to take very small values (and thus become approximately zero).

Once we finished the sampling, we define the measurement signal $\mathbf{i}_m^j(t)$ in the j-th vehicle as

$$\mathbf{i}_m^j(t) = \left(\mathbf{I} + \mathbf{D}_m^j\right)\left(1 + d_h^j\right)\mathbf{i}_s(t) + \left(\mathbf{I} + \mathbf{D}_m^j\right)\mathbf{o}_h^j + \mathbf{o}_m^j, \tag{21}$$

where d_h^j, $\mathbf{o}_h^j$, $\mathbf{D}_m^j$ and $\mathbf{o}_m^j$ are the sampled parameters.

On top of the above listed sampling conditions on the individual vehicle level, we make sure that on the fleet-level, the following two conditions are satisfied:

- The fleet is **balanced**, i.e., the ratio of EVs affected by hardware faults to those which are not is around 1:1. Thus, we force $\mathbf{o}_h$ and $\mathbf{d}_h$ to be zero for half of the vehicles in the artificial fleet. We use this trick because for a larger K, and a smaller sparsity level S, almost all EVs would be affected by hardware faults.

1. In our implementation, we utilize a slightly different parametrization, namely by only preserving the first $K - 1$ components of $\mathbf{o}_h$ as unknown parameters, and setting $o_{h,K} = -\sum_{j=1}^{K-1} o_{h,j}$. This constraint is then satisfied automatically.

According to our experiences, this would be very unrealistic. Especially, if the fleet contains EVs of young age with a low amount of vehicle miles traveled.

- The fleet is **symmetric**, i.e., we have for all $t \in \{1, \dots, T\}$

$$\mathbf{i}_s(t) = \frac{1}{N} \sum_{j=1}^{N} \mathbf{i}_m^j(t). \tag{22}$$

This condition makes sure that Assumption 1 is nearly satisfied.[2] It is enforced by creating for each vehicle in the fleet with parameters $(d_h, \mathbf{o}_h, \mathbf{D}_m, \mathbf{o}_m)$, three other vehicles with parameters $(-d_h, -\mathbf{o}_h, -\mathbf{D}_m, -\mathbf{o}_m)$, $(d_h, \mathbf{o}_h, -\mathbf{D}_m, \mathbf{o}_m)$, and $(-d_h, -\mathbf{o}_h, \mathbf{D}_m, -\mathbf{o}_m)$, respectively. This means that we assume N to be divisible by 4.

C. SIMULATION MODELS

In the previous two subsections, we define the ground truth signal $\mathbf{i}_s(t)$ and the measurement signals $\mathbf{i}_m^j(t)$ for all $j \in \{1, \dots, N\}$ and $t \in \{1, \dots, T\}$. The only step left is to define the simulation signals $\mathbf{i}_{sim}^j(t)$ for all $j \in \{1, \dots, N\}$. To do so, we differentiate between two methods:

- *State-Space Simulation:* For each j, we split the available data into two parts. We define $T_1 = \lfloor 0.8T \rfloor$ and use the first part $\{\mathbf{i}_m^j(t)\}_{1 \le t \le T_1}$ to train and validate a State-Space Model which is able to simulate the dynamics of the j-th vehicle. By simulating the model on the remaining 20% of the available data,[3] we obtain $\{\mathbf{i}_{sim}^j(t)\}_{T_1+1 \le t \le T}$.
- *Artificial Simulation:* Alternatively, we assume that the error $\mathbf{e}_{sim}^j = \mathbf{i}_{sim}^j - \mathbf{i}_m^j$ is distributed as $\mathcal{N}(\mathbf{0}, \sigma_{sim}^2 \mathbf{I})$ for some variance σ_{sim}^2 and define for $j \in \{1, \dots, N\}$ and $t \in \{T_1 + 1, \dots, T\}$

$$\mathbf{i}_{sim}^j(t) = \mathbf{i}_m^j(t) + \mathbf{e}_{sim}^j(t) \quad \text{where} \quad \mathbf{e}_{sim}^j(t) \sim \mathcal{N}\left(\mathbf{0}, \sigma_{sim}^2 \mathbf{I}\right). \tag{23}$$

While our method originally relies on the state-space simulation as described in [3], the second method allows us to provide more general results by investigating the effect of σ_{sim}^2 on the overall performance of our method. Besides that, it makes it easier to conduct experiments, especially for larger values of N.

D. EXPERIMENTS

Following the previous sections, each experiment consists of the following steps:

1) As described in Section IV-A, generate for all K currents and time steps $t \in \{1, \dots, T\}$ an artificial ground truth $\mathbf{i}_s(t)$.
2) Depending on $\mathbf{i}_s(t)$, $j \in \{1, \dots, N\}$ and $t \in \{1, \dots, T\}$, create a balanced artificial random fleet of N vehicles as described in Section IV-B with sparsity level S to obtain $\mathbf{i}_m^j(t)$.
3) Choose one of the simulation methods described in Section IV-C to obtain the simulated currents $\{\mathbf{i}_{sim}^j(t)\}_{T_1+1 \le t \le T}$.
4) Based on $\{\mathbf{i}_{sim}^j(t)\}_{T_1+1 \le t \le T}$, estimate the ground truth as described in Section III-A according to

$$\tilde{\mathbf{i}}_s(t) = \frac{1}{N} \sum_{j=1}^{N} \mathbf{i}_{sim}^j(t). \tag{24}$$

5) Based on $\{\tilde{\mathbf{i}}_s(t)\}_{T_1+1 \le t \le T}$ and $\{\mathbf{i}_{sim}^j(t)\}_{T_1+1 \le t \le T}$, **classify** whether the j-th vehicle is affected by **measurement** and/or **hardware faults** as described in Section III-B. Depending on the experiment, we either utilize the simple thresholding classifier with parameter δ_h, or a previously trained decision tree classifier. In the latter case, the classifier is trained on another data set $\mathcal{D}_{train}$. This data set is created artificially in the same way (i.e., using the same sampling scheme and parameters) as $\mathcal{D}$.
6) For each vehicle in the fleet, follow the descriptions in Section III-C to recover the parameters $(d_h, \mathbf{o}_h, m, \mathbf{o}_m)$.
7) Calculate the recovery rate, i.e., the ratio of cases for which $\mathbf{D}_m$ and $\mathbf{o}_m$ have been recovered with a relative error below 10%.[4] The reason we only focus on $\mathbf{D}_m$ and $\mathbf{o}_m$ is that we do not need the other parameters to be able to correct the current via the inverse mapping

$$\mathbf{i}_m \mapsto (\mathbf{I} + \mathbf{D}_m)^{-1}(\mathbf{i}_m - \mathbf{o}_m). \tag{25}$$

V. RESULTS

In this section, we evaluate our approach in an end-to-end fashion. Therefor, we run the experiment from Section IV-D multiple times with different parameter combinations (i.e., values of the number of currents K, the fleet size N, the sparsity level S, etc.). By fixing all parameters but a few, we investigate in a number of experiment series the effect of the variable parameters on the overall achieved performance as measured by the recovery rate defined above. Due to the large number of parameters involved, we will thereby restrict our analysis to the following effects.

A. EFFECT OF THE SIMULATION NOISE VARIANCE

All the steps of the proposed approach rely on simulated signals, so that a too large simulation error is likely to cause the recovery to fail. To avoid such a failure, it is important to quantify how *good* the simulation should be for the algorithm to achieve a sufficiently high recovery rate. In other words, we investigate the stability of our approach against simulation noise. For this investigation, we fix $N = 1000$, $K = 2$,

2. Note that Assumption 1 uses the simulated currents and not the measured currents.

3. For both training and simulation, we also use the corresponding input vector $\{\mathbf{u}(t)\}_{1 \le t \le T}$ which is available from the original data set used to build the artificial ground truth (see Section IV-A).

4. For our purposes, it is sufficient to assume cases with a relative error below 10% as recovered. Of course, the reader is free to choose a value depending on the individual use case.

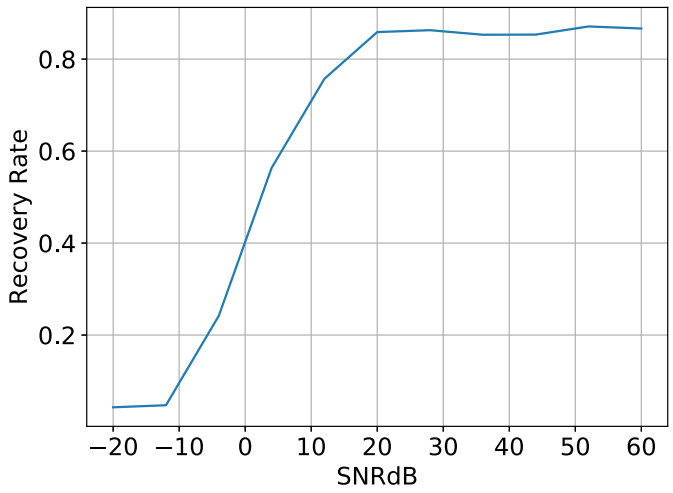


FIGURE 7. The recovery rate is plotted against the signal-to-noise ratio *SNRdB*.

and $S = 60$. For our experiments, we use the signal-to-noise ratio *SNRdB* to measure the goodness of a simulation. We run for the experiment 10 times for several signal-to-noise ratios in the interval $[-20, 60]$ with

$$\sigma_{sim}^2 = 10^{-SNRdB/10} \cdot P, \tag{26}$$

where P is the average power of $i_{s,1}(t)$,[5] i.e., $P = T^{-1} \sum_{t=1}^{T} i_{s,1}(t)^2$. Note that in order to isolate the investigated effect from other noise sources, we enforce a symmetric fleet and a perfect classifier. A non-symmetric fleet would introduce an additional noise in the ground truth estimation (see (4)), and false negatives in the classification would introduce a systematic error in the parameter estimation.[6]

In Figure 7, we plot the average recovery rate (across the 10 repetitions) as a function of *SNRdB*. We see that an SNR above around 15 dB is necessary for the algorithm to have a recovery rate above 80%, which we consider to be a sufficiently mild requirement on the simulation model. In fact, the State-Space Models introduced in our previous work [3] largely satisfy this requirement.

B. EFFECT OF THE CLASSIFIER

As discussed in Section III-C, classification errors in general, and false negatives in particular, are harmful to the recovery rate. To investigate which of the two suggested classifiers (simple thresholding rule vs. decision tree) produces better results, we conduct the following experiment. We set $N = 1000, K = 2$ and $S = 60$ and run the experiment from Section IV-D 10 times for each classifier with $SNRdB = 20$ in (26). Thereby, on the one hand, we set for the thresholding classifier $\delta_h = \tilde{\delta}_h \sigma_\eta$, where $\tilde{\delta}_h$ is empirically set to $1/3$, and

5. Because $K = 2$, we have $i_{s,1}(t) = i_{s,2}(t)$, so that P is also the average power of $i_{s,2}(t)$.

6. For false negatives, the classifier wrongly decides there are no hardware faults. All parameters relating to hardware faults, i.e., d_h and $\mathbf{o}_h$, are wrongly set to 0, which leads to systematic errors in the estimation of $\mathbf{d}_m$ and $\mathbf{o}_m$.

TABLE 1. Results of the second experiment for simple thresholding, the decision tree classifier and the perfect classifier.

	Simple thresholding	Decision tree	Perfect
Mean f_2 score	0.68	**0.84**	1.00
Mean recovery rate	0.84	**0.86**	0.91
Mean complexity	0.93	**0.41**	0.40

σ_η is the standard deviation of the decision criterion across the fleet (see (7)), i.e.,

$$\sigma_\eta = \sqrt{\frac{1}{N-1} \sum_{j=1}^{N} \left(\eta_j - \bar{\eta}_j\right)} \quad \text{with} \quad \bar{\eta}_j = \sum_{j=1}^{N} \eta_j. \tag{27}$$

On the other hand, we train the decision tree on an artificial fleet of 3,000 vehicles. The fleet is created using the same hyper-parameters as the ones utilized to create the primary fleet (i.e., with the same sparsity level S and parameter distributions a and b). We further restrict the tree depth to 6 to prevent overfitting.

On top of the two suggested classifiers, we include the results obtained using a perfect classifier as benchmark. For the evaluation, we use the following three metrics:

- The f_2 score as a performance measure of each classifier. We chose this score since we consider false negatives to be more harmful than false positives, and thus consider recall to be more important than precision.[7]
- The recovery rate as a measure of the end-to-end performance.
- The percentage of cases for which the optimization procedure in (18) is executed (i.e., the percentage of cases classified as displaying hardware errors) as a complexity measure.

Table 1 summarizes the results of this experiment. We see that the decision tree classifier outperforms the simple thresholding rule on all evaluation criteria. In particular, thanks to its better f_2 score, the decision tree classifier is able to reduce the complexity score to only 0.41, so that the classification step is fulfilling its originally conceived target. Note that this gain in computational resources does compensate the resources needed to train the tree in the first place.

C. EFFECT OF THE SPARSITY LEVEL

The sparsity constraint is paramount to make the originally ill-posed problem well-posed again. We therefore want to evaluate our method for various values of S, and find out *how much sparsity* is actually required to obtain a high recovery rate. To do this, we run our algorithm in an end-to-end fashion using the same setting as the last experiment, only this time fixing the classifier to be a decision tree, and letting S vary in $\{20, 40, 60, 80\}$. The results are summarized in Figure 8.

7. This corresponds to the particular choice of $\beta = 2$ in the more general f_β score defined as $f_\beta = (1 + \beta^2)pr/(\beta^2 p + r)$, which is a weighted harmonic mean of recall and precision, where the weight for recall (r) is β^2 times that of precision (p).

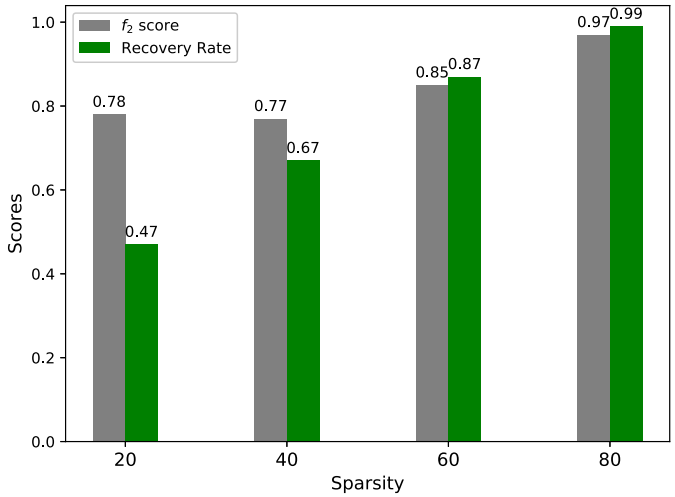


FIGURE 8. Classification and recovery rates for different sparsity levels.

TABLE 2. Results of the end-to-end evaluation using state-space models.

Sparsity	S=20	S=40	S=60	S=80
Mean f_2 score	0.76	0.75	0.91	0.91
Recovery rate ($< 10\%$)	0.45	0.49	0.75	0.96
Recovery rate ($< 25\%$)	0.63	0.71	0.85	0.98

D. OVERALL EVALUATION

As last experiment, we evaluate our method in an end-to-end fashion using the state-space approach suggested in [3] as concrete simulation model. We thereby set $K = 2$ and $N = 500$, enforce the fleets to be symmetric and balanced, and use pre-trained decision trees for the classification step. Each tree is trained on an artificial data set created using the artificial simulation method with $SNRdB = 20$. When executing this for various values of S, we obtain the results in Table 2.

VI. DISCUSSION

Our approach for the correction of measurement faults in the power train of EVs consists of 3 steps. First, we estimate the ideal ground truth. Second, we classify hardware and measurement faults. Finally, the measurement faults are recovered based on the classification result.

To evaluate our approach, we execute 4 experiments. In the first experiment, we show that the recovery rate increases for lower simulation errors. Our previously implemented State-Space Models satisfy the experimental requirement of 15 dB needed to achieve a high recovery rate ($> 80\%$, see Figure 7). It is interesting that the recovery rate improves just marginally at higher dB levels. This means that we need a relatively accurate simulation model but do not require highest accuracy. Thus, our approach is capable to handle simulation errors to a certain extent.

Our second experiment shows that decision trees are superior to the thresholding rule. There are several reasons for the decision trees' advantages. First, they are trained on the data before they are deployed in contrast to rules which are set manually. Second, besides simple thresholding, they are able to learn complex decisions rules and third, they take all defined features as input. However, decision trees also have some drawbacks. They require training which in turn requires the creation of an artificial fleet. For the fleet's creation, we need to state distributional assumptions about the fleet. These assumptions might add uncertainty to our data and lead to wrong decisions. Overall, even if the improvement of decision trees to the recovery rate is relatively small (0.86 vs. 0.84 as can be seen in Table 1), we still find them better as they achieve a better classification rate (0.84 vs. 0.68 as can be seen in Table 1), which, as discussed in the end of Section III, reduces the required computational power.

The main insight of our third experiment is that the sparsity assumption is the crucial influencing factor on the recovery rate. The recovery rate can be decreased to 47% by low sparsity levels. This behavior is comprehensible since the recovery algorithm is conceived to work with sparse vectors and thus to minimize the 1-norm. The goal of sparsity constraints is to deal with the ill-posedness of problems. Assuming low sparsity means to deal with highly ill-posed problems and low recovery.

From the fourth experiment, we learn that the recovery rate is lower (e.g., 75% instead of 86% for $S = 60$) for State-Space models instead of artificial simulation (see Table 1 and Table 2). This is surprising for us because State-Space Models also fulfill the dB requirement of the first experiment. A possible reason for the lower recovery rate might be that the errors induced by the State-Space Models satisfy the dB requirement for the standard deviation but are not normally distributed. This might lead to some correlations and decrease the accuracy of the regression in the first recovery step. Further research into the exact reasons for the lower recovery rate of State-Space Models is up to our future work.

Regarding the whole work described in this article, we realize that Assumption 2 is crucial for the results. Lower sparsities induce worse results. This means that our algorithm is sensitive to the previously stated assumption. We expect this drawback to be solved by stabilizing the algorithm with respect to simulation errors. Overall, if the assumptions are fulfilled, the results look very promising. The good results with high recovery rates serve as basis for our future work. Furthermore, we want to replace the simple linear error model by more complex non-linear models.

VII. CONCLUSION

This article presents an advanced fleet-based framework to correct measurement faults in the power trains of EVs. Through a comparison with the mean behavior of the fleet, we are able to classify whether a certain vehicle suffers from significant hardware errors. Then, based on the classification result, we use a combination of linear regression and convex sparse optimization to recover the parameters defining measurement errors. Using relatively mild and realistic assumptions, we thereby achieve a high recovery rate of up to 90%. Overall, our framework is able to correct measurement

faults in a completely automated way, and without additional sensors or computational power on-board of the EV.

ABBREVIATIONS

ECU	Electronic Control Unit
EV	Electric Vehicle
HV	High Voltage
HVB	High Voltage Battery
PCA	Principal Component Analysis
QBP	Quadratic Basis Pursuit
RMSE	Root Mean Square Error

REFERENCES

[1] P. Komarnicki, J. Haubrock, and Z. A. Styczynski, "Elektrische Komponenten des E-Kfz," in *Elektromobilität und Sektorenkopplung*. Heidelberg, Germany: Springer, 2018, pp. 61–109. [Online]. Available: https://link.springer.com/chapter/10.1007%2F978-3-662-56249-9_3

[2] J. Pfeiffer, X. Wu, and A. Ayadi, "Evaluation of three different approaches for automated time delay estimation for distributed sensor systems of electric vehicles," *Sensors*, vol. 20, no. 2, p. 351, Jan. 2020. [Online]. Available: https://www.mdpi.com/1424-8220/20/2/351

[3] J. Pfeiffer, P. Wolf, and R. Pereira, "A fleet-based machine learning approach for automatic detection of deviations between measurements and reality," in *Proc. IEEE Intell. Veh. Symp. (IV)* Paris, France, Jun. 2019, pp. 2086–2092. [Online]. Available: https://ieeexplore.ieee.org/document/8813858/

[4] N. K. Malakar *et al.*, "Estimation and bias correction of aerosol abundance using data-driven machine learning and remote sensing," in *Proc. IEEE Conf. Intell. Data Understand. (CIDU)*, Boulder, CO, USA, 2012, pp. 24–30.

[5] S. Koos, J.-B. Mouret, and S. Doncieux, "The transferability approach: Crossing the reality gap in evolutionary robotics," *IEEE Trans. Evol. Comput.*, vol. 17, no. 1, pp. 122–145, Feb. 2013.

[6] Z. Zhao, Y.-G. Sun, and J. Zhang, "Fault detection and diagnosis for sensor in an aero-engine system," in *Proc. IEEE Chin. Control Decis. Conf. (CCDC)*, Yinchuan, China, 2016, pp. 2977–2982.

[7] N. Jakobi, P. Husbands, and I. Harvey, "Noise and the reality gap: The use of simulation in evolutionary robotics," in *Proc. 3rd Eur. Conf. Artif. Life*, vol. 929, 1995, pp. 704–720.

[8] J. Lu, J. Huang, and F. Lu, "Sensor fault diagnosis for aero engine based on online sequential extreme learning machine with memory principle," *Energies*, vol. 10, no. 1, p. 39, 2017.

[9] S. Ren, Q. Liu, and X. Liu, "Heavy DC current calibration method based on high-precision current adder," *IEEE Trans. Instrum. Meas.*, vol. 63, no. 12, pp. 3039–3044, Dec. 2014. [Online]. Available: http://ieeexplore.ieee.org/document/6867367/

[10] T. Kobayashi and D. L. Simon, "Evaluation of an enhanced bank of Kalman filters for in-flight aircraft engine sensor fault diagnostics," *J. Eng. Gas Turb. Power*, vol. 127, no. 3, p. 497, 2005.

[11] H. Ohlsson, A. Y. Yang, R. Dong, M. Verhaegen, and S. S. Sastry, "Quadratic basis pursuit," Jan. 2013. [Online]. Available: http://arxiv.org/abs/1301.7002.

[12] J. Pfeiffer and X. Wu, "Automated time delay estimation for distributed sensor systems of electric vehicles," in *Proc. 10th IEEE Int. Conf. Intell. Data Acquisition Adv. Comput. Syst. Technol. Appl. (IDAACS)*, Metz, France, Sep. 2019, pp. 609–614. [Online]. Available: https://ieeexplore.ieee.org/document/8924330/

AHMED AYADI received the B.Sc. degree in electrical engineering and information technology and the M.Sc. degree in management from the Technical University of Munich, Munich, Germany, in 2016 and 2018, respectively, where he is currently pursuing the M.Sc. degree in electrical engineering and information technology. His current focus is on machine learning methods and theory.

JAKOB PFEIFFER received the B.Sc. degree in informatics and the M.Sc. degree in robotics, cognition, intelligence from the Technical University of Munich, Munich, Germany, where he is currently pursuing the Ph.D. degree with the Chair for Data Processing, Department of Electrical and Computer Engineering and participating in the ProMotion program of the BMW Group, Munich. In his current work in the field of machine learning, he is contributing to minimizing the differences between measured and real values in the power trains of EVs.

5 Conclusion

In this thesis, I propose a self-learning correction of measurement deviations of HV currents in the power trains of EVs. The correction distinguishes between measurement deviations caused by sensor inaccuracies and by time delays. It works with the sensor data available in modern production vehicles and can be executed on the processing resources of automotive ECUs. To keep the additional effort for applicators as low as possible, I propose and evaluate several methods from the field of Machine Learning. These algorithms are able to adjust themselves to the data and require no or at least only minimal manual parametrization.

The variety of Machine Learning methods evaluated in the scope of this thesis in chapters 3 and 4 shows that it depends strongly on the individual problem which algorithm is suited best to achieve the required result. As you can see especially from section 3.2, sometimes Machine Learning methods are not the best choice and naive but simple methods can achieve comparable results with less effort.

The proposed measurement correction allows to minimize the additional offsets on the battery protection limits introduced in section 1.1. The smaller offsets allow to increase the efficiency as well as the performance of the power train of an EV. For example, the time delay correction presented in section 3.1 reduces the measurement deviation from 25 % of the maximum current to below 5 %. This means that the offsets on the battery protection limits can be reduced by 80 % and the maximum power of the EV can be increased by approximately 27 % while at the same time increasing the efficiency due to higher recuperation. Additionally, the fleet-based measurement correction proposed in chapter 4 allows to correct drift and bias faults in the HV current measurements with a recovery rate of up to 90 %.

The automated, self-learning measurement correction shows that Machine Learning methods allow to replace application by functional development to a certain extent. However, as especially section 4.2 shows, manual application is still required to retrieve optimal and efficient results. Thus, I would rather conclude that Machine Learning transforms the process of application in the automotive software than replacing it completely. The task of applicators might change to adjusting models in such a way that they adapt themselves optimally to the input data in a fast way. The models might be extended to Machine Learning methods instead of the controller models which are common in the electric power train nowadays.

Acronyms

ABS	Anti-lock Braking System
ARIMA	Auto-Regressive Integrated Moving Average
ARMA	Auto-Regressive Moving Average
BATS	Box-cox transformation, ARMA residuals, Trend and Seasonality
BEV	Battery Electric Vehicle
CAN	Controller Area Network
DSC	Dynamic Stability Control
ECU	Electronic Control Unit
EM	Electric Machine
EV	Electric Vehicle
GPS	Global Positioning System
HV	High Voltage
HVB	High Voltage Battery
ICEV	Internal Combustion Engine Vehicle
LIDAR	Light Detecting And Ranging
LSTM	Long Short-Term Memory Neural Network
LV	Low Voltage
MOS-ELM	Online Sequential Extreme Learning Machine with Memory principle
MOESP	Multivariable Output Error State Space
N4SID	Numerical State Space Subspace System Identification
PHEV	Plug-in Hybrid Electric Vehicle
QBP	Quadratic Basis Pursuit
RADAR	Radio Detecting And Ranging
RMSE	Root Mean Square Error
SoC	State of Charge
TBATS	Trigonometric seasonal, Box-cox transformation, ARMA residuals, Trend and Seasonality
TDE	Time Delay Estimation

Bibliography

[1] Anthony Babin, Nassim Rizoug, Tedjani Mesbahi, David Boscher, Zouheir Hamdoun, and Cherif Larouci. Total Cost of Ownership Improvement of Commercial Electric Vehicles Using Battery Sizing and Intelligent Charge Method. *IEEE Transactions on Industry Applications*, 54(2):1691–1700, March 2018.

[2] Michael Beuschel. *Neuronale Netze zur Diagnose und Tilgung von Drehmomentschwingungen am Verbrennungsmotor*. Dissertation, Technical University of Munich, Munich, 2000.

[3] Maximilian Cussigh and Thomas Hamacher. Optimal Charging and Driving Strategies for Battery Electric Vehicles on Long Distance Trips: a Dynamic Programming Approach. In *2019 IEEE Intelligent Vehicles Symposium (IV)*, pages 2093–2098, Paris, France, June 2019. IEEE.

[4] Alysha M. De Livera, Rob J. Hyndman, and Ralph D. Snyder. Forecasting Time Series With Complex Seasonal Patterns Using Exponential Smoothing. *Journal of the American Statistical Association*, 106(496):1513–1527, December 2011.

[5] Raimund Ellinger. Virtuelle Antriebsstrangentwicklung. In Ulrich Seiffert and Gotthard Rainer, editors, *Virtuelle Produktentstehung für Fahrzeug und Antrieb im Kfz*, pages 154–171. Vieweg+Teubner, Wiesbaden, 2008.

[6] Franz Froschhammer. *Identifikation und Kompensation von Sensorungenauigkeiten für die drehzahlgestützte On-Board-Diagnose von Verbrennungsmotoren*. Dissertation, Technical University of Munich, Munich, 2002.

[7] Robert Bosch GmbH. Motormanagement ME-Motronic. In Robert Bosch GmbH, editor, *Ottomotor-Management*. Vieweg+Teubner, Wiesbaden, 1998.

[8] Takoi K. Hamrita, Bonnie S. Heck, and A. P. Sakis Meliopoulos. On-line correction of errors introduced by instrument transformers in transmission-level steady-state waveform measurements. *IEEE Transactions on Power Delivery*, 15(4):1116–1120, October 2000.

[9] Kichun Jo, Keonyup Chu, and Myoungho Sunwoo. GPS-bias correction for precise localization of autonomous vehicles. In *IEEE Intelligent Vehicles Symposium (IV)*, pages 636–641, 2013.

[10] Sören Kammel, Julius Ziegler, Benjamin Pitzer, Moritz Werling, Tobias Gindele, Daniel Jagzent, Joachim Schröder, Michael Thuy, Matthias Goebl, Felix von Hundelshausen, Oliver Pink, Christian Frese, and Christoph Stiller. Team AnnieWAY's autonomous system for the 2007 DARPA Urban Challenge. *Journal of Field Robotics*, 25(9):615–639, September 2008.

[11] Przemyslaw Komarnicki, Jens Haubrock, and Zbigniew A. Styczynski. *Elektromobilität und Sektorenkopplung: Infrastruktur- und Systemkomponenten*. SpringerLink Bücher. Springer Vieweg, Berlin, 2018.

[12] Karl Koscher, Alexei Czeskis, Franziska Roesner, Shwetak Patel, Tadayoshi Kohno, Stephen Checkoway, Damon McCoy, Brian Kantor, Danny Anderson, Hovav Shacham, and Stefan Savage. Experimental Security Analysis of a Modern Automobile. In *2010 IEEE Symposium on Security and Privacy*, pages 447–462, Oakland, CA, USA, 2010. IEEE.

[13] Alexander Kowallik. *Performancesteigerung im E-Antriebssystem durch szenariobasierte thermische Konditionierung des Hochvoltspeichers*. Dissertation, Technical University of Darmstadt, Darmstadt, 2020.

[14] Yoshiaki Kuwata, Justin Teo, Sertac Karaman, Gaston Fiore, Emilio Frazzoli, and Jonathan How. Motion Planning in Complex Environments Using Closed-loop Prediction. In *AIAA Guidance, Navigation and Control Conference and Exhibit*, Honolulu, Hawaii, August 2008. American Institute of Aeronautics and Astronautics.

[15] John Leonard, Jonathan How, Seth Teller, Mitch Berger, Stefan Campbell, Gaston Fiore, Luke Fletcher, Emilio Frazzoli, Albert Huang, Sertac Karaman, Olivier Koch, Yoshiaki Kuwata, David Moore, Edwin Olson, Steve Peters, Justin Teo, Robert Truax, Matthew Walter, David Barrett, Alexander Epstein, Keoni Maheloni, Katy Moyer, Troy Jones, Ryan Buckley, Matthew Antone, Robert Galejs, Siddhartha Krishnamurthy, and Jonathan Williams. A perception-driven autonomous urban vehicle. *Journal of Field Robotics*, 25(10):727–774, October 2008.

[16] Yuecheng Li, Hongwen He, Jiankun Peng, and Hong Wang. Deep Reinforcement Learning-Based Energy Management for a Series Hybrid Electric Vehicle Enabled by History Cumulative Trip Information. *IEEE Transactions on Vehicular Technology*, 68(8):7416–7430, August 2019.

[17] Junjie Lu, Jinquan Huang, and Feng Lu. Sensor Fault Diagnosis for Aero Engine Based on Online Sequential Extreme Learning Machine with Memory Principle. *Energies*, 10(1):39, January 2017.

[18] Dario Pevec, Jurica Babic, Arthur Carvalho, Yashar Ghiassi-Farrokhfal, Wolfgang Ketter, and Vedran Podobnik. Electric Vehicle Range Anxiety: An Obstacle for the

Personal Transportation (R)evolution? In *2019 4th International Conference on Smart and Sustainable Technologies (SpliTech)*, pages 1–8, Split, Croatia, June 2019. IEEE.

[19] Jakob Pfeiffer, Peter Wolf, and Roberto Pereira. A Fleet-Based Machine Learning Approach for Automatic Detection of Deviations between Measurements and Reality. In *2019 IEEE Intelligent Vehicles Symposium (IV)*, pages 2086–2092, Paris, France, June 2019. IEEE.

[20] Shiyan Ren, Qingxin Liu, and Xiaojun Liu. Heavy DC Current Calibration Method Based on High-Precision Current Adder. *IEEE Transactions on Instrumentation and Measurement*, 63(12):3039–3044, December 2014.

[21] Thomas Salcher. *Optimierte Betriebsstrategie hybrider Antriebssysteme für den Serieneinsatz*. Dissertation, Technical University of Munich, Munich, 2013.

[22] Jörg Schäuffele and Thomas Zurawka. *Automotive Software Engineering: Grundlagen, Prozesse, Methoden und Werkzeuge effizient einsetzen*. Praxis. Vieweg+Teubner, Wiesbaden, 4th edition, 2010.

[23] Tobias Straub, Mandy Nagy, Maxim Sidorov, Leonardo Tonetto, Michael Frey, and Frank Gauterin. Energetic Map Data Imputation: A Machine Learning Approach. *Energies*, 13(4):982, February 2020.

[24] Thomas Vietor and Carsten Stechert. Produktarten zur Rationalisierung des Entwicklungs- und Konstruktionsprozesses. In Jörg Feldhusen and Karl-Heinrich Grote, editors, *Pahl/Beitz Konstruktionslehre*, pages 817–871. Springer Berlin Heidelberg, Berlin, Heidelberg, 2013.

[25] Xiebing Wang, Kai Huang, Long Chen, and Alois Knoll. h²ECU: A High-Performance and Heterogeneous Electronic Control Unit for Automated Driving. *IEEE Micro*, 38(5):53–62, September 2018.

[26] Peter Wolf, Artur Mrowca, Tam Thanh Nguyen, Bernard Baker, and Stephan Gunnemann. Pre-ignition Detection Using Deep Neural Networks: A Step Towards Data-driven Automotive Diagnostics. In *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, pages 176–183, Maui, HI, November 2018. IEEE.

[27] Rui Xiong, Jiayi Cao, Quanqing Yu, Hongwen He, and Fengchun Sun. Critical Review on the Battery State of Charge Estimation Methods for Electric Vehicles. *IEEE Access*, 6:1832–1843, 2018.

Appendix

Related Patent Applications

The contribution to the state of the art from chapter 2 and the implementation of my co-authors and me in chapters 3 and 4 lead to five patent applications. The patent applications are presented more detailed below. Additionally, I registered another patent application protecting the idea of charging EVs while being transported on trains. However, as this patent application does not fit the scope of the thesis, it is not listed below.

Method for Computer-Aided Evaluation of a Measurement of an Electrical Unit in a High-Voltage On-Board Network of a Predetermined Electrically Driven Motor Vehicle

Inventors: Jakob Pfeiffer, Peter Wolf, Roberto Pereira
Official file number: 10 2018 132 658.0
Registration date: December 12, 2018

This patent application is created in the context of the paper from section 4.1. It protects the idea of fleet-based measurement deviation detection and the resulting correction with the help of Machine Learning. The Machine Learning methods are used to learn measurement models on-board of individual EVs. The deviation correction is performed in the backup with the help of the classifier proposed in the paper.

Method, Device, Computer Program and Computer Program Product for Processing Measurement Data Sets (Linear Regression)

Inventor: Jakob Pfeiffer
Official file number: 10 2019 106 461.9
Registration date: March 14, 2019

This patent application results from the work in the context of the papers printed in section 3.1. It protects the idea of correcting measurement deviations caused by time delays. Especially the measurement correction based on Linear Regression is protected here.

Method, Device, Computer Program and Computer Program Product for Processing Measurement Data Sets (Error Measure Minimization)

Inventor: Jakob Pfeiffer
Official file number: 10 2019 108 328.1
Registration date: March 29, 2019

Like the patent application above, this one results from the work in the context of the papers printed in section 3.1. Its focus is on protecting the idea of correcting measurement deviations caused by time delays by minimizing an error measure, for example

the RMSE. The RMSE approach is a predecessor of the Variance Minimization approach presented in the papers.

Please note that in addition to this patent application, Xuyi Wu and Ahmed Ayadi registered another patent application protecting the stabilization for this approach proposed in the second paper in section 3.1.

Method for Computer-Aided Evaluation of Measurements of Electrical Currents in a High-Voltage On-Board Network of a Predetermined Electrically Driven Motor Vehicle

Inventors: Jakob Pfeiffer, Ahmed Ayadi
Official file number: 10 2019 135 022.0
Registration date: December 18, 2019

This patent application results from the work in the context of the paper presented in section 4.2. It protects the correction of measurement deviation caused by measurement inaccuracies. Additionally to the patent registration from 5, it protects the new classifier and the measurement correction based on sparsity constraints.

System, Method and Computer Program for Determining Estimated Sensor Data

Inventors: Jakob Pfeiffer, Mohamed Ali Razouane
Official file number: EP20171463.1
Registration date: April 17, 2020

Resulting from the work of the context presented in section 3.2, this patent application protects the idea of predictive reconstruction of delayed measurements with time series prediction algorithms.

European Research Award

The research topic and the presented solutions of this work have been awarded with the third place in the category *Road* of the *TRA VISIONS Young Researcher Competition 2020*. For further information on TRA VISIONS and the related transport research award, I kindly refer the interested reader to `https://www.travisions.eu/TRAVisions/`.

Copyrights

In the following, I print the copyright statements allowing me to print all the papers in the scope of this thesis. Regarding the second, extended paper from section 3.1 and the paper from section 4.2 please consider that these are published under the Creative Commons Attribution 4.0 International License. This license explicitly authorizes the copying and distribution of the papers. You can find the detailed license under `https://creativecommons.org/licenses/by/4.0/`. The papers in this thesis are printed as copy of the original papers without any changes.

Automated Time Delay Estimation for Distributed Sensor Systems of Electric Vehicles

Conference Proceedings:
2019 10th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS)

Author: Jakob Pfeiffer; Xuyi Wu

Publisher: IEEE

Date: 18-21 Sept. 2019

Thesis / Dissertation Reuse

The IEEE does not require individuals working on a thesis to obtain a formal reuse license, however, you may print out this statement to be used as a permission grant:

Requirements to be followed when using any portion (e.g., figure, graph, table, or textual material) of an IEEE copyrighted paper in a thesis:

1) In the case of textual material (e.g., using short quotes or referring to the work within these papers) users must give full credit to the original source (author, paper, publication) followed by the IEEE copyright line © 2011 IEEE.
2) In the case of illustrations or tabular material, we require that the copyright line © [Year of original publication] IEEE appear prominently with each reprinted figure and/or table.
3) If a substantial portion of the original paper is to be used, and if you are not the senior author, also obtain the senior author's approval.

Requirements to be followed when using an entire IEEE copyrighted paper in a thesis:

1) The following IEEE copyright/ credit notice should be placed prominently in the references: © [year of original publication] IEEE. Reprinted, with permission, from [author names, paper title, IEEE publication title, and month/year of publication]
2) Only the accepted version of an IEEE copyrighted paper can be used when posting the paper or your thesis on-line.
3) In placing the thesis on the author's university website, please display the following message in a prominent place on the website: In reference to IEEE copyrighted material which is used with permission in this thesis, the IEEE does not endorse any of [university/educational entity's name goes here]'s products or services. Internal or personal use of this material is permitted. If interested in reprinting/republishing IEEE copyrighted material for advertising or promotional purposes or for creating new collective works for resale or redistribution, please go to http://www.ieee.org/publications_standards/publications/rights/rights_link.html to learn how to obtain a License from RightsLink.

If applicable, University Microfilms and/or ProQuest Library, or the Archives of Canada may supply single copies of the dissertation.

BACK CLOSE WINDOW

Time Series Prediction for Measurements of Electric Power Trains

Conference Proceedings:
2020 Forum on Integrated and Sustainable Transportation Systems (FISTS)

Author: Jakob Pfeiffer

Publisher: IEEE

Date: 3 Nov. 2020

Thesis / Dissertation Reuse

The IEEE does not require individuals working on a thesis to obtain a formal reuse license, however, you may print out this statement to be used as a permission grant:

Requirements to be followed when using any portion (e.g., figure, graph, table, or textual material) of an IEEE copyrighted paper in a thesis:

1) In the case of textual material (e.g., using short quotes or referring to the work within these papers) users must give full credit to the original source (author, paper, publication) followed by the IEEE copyright line © 2011 IEEE.
2) In the case of illustrations or tabular material, we require that the copyright line © [Year of original publication] IEEE appear prominently with each reprinted figure and/or table.
3) If a substantial portion of the original paper is to be used, and if you are not the senior author, also obtain the senior author's approval.

Requirements to be followed when using an entire IEEE copyrighted paper in a thesis:

1) The following IEEE copyright/ credit notice should be placed prominently in the references: © [year of original publication] IEEE. Reprinted, with permission, from [author names, paper title, IEEE publication title, and month/year of publication]
2) Only the accepted version of an IEEE copyrighted paper can be used when posting the paper or your thesis on-line.
3) In placing the thesis on the author's university website, please display the following message in a prominent place on the website: In reference to IEEE copyrighted material which is used with permission in this thesis, the IEEE does not endorse any of [university/educational entity's name goes here]'s products or services. Internal or personal use of this material is permitted. If interested in reprinting/republishing IEEE copyrighted material for advertising or promotional purposes or for creating new collective works for resale or redistribution, please go to http://www.ieee.org/publications_standards/publications/rights/rights_link.html to learn how to obtain a License from RightsLink.

If applicable, University Microfilms and/or ProQuest Library, or the Archives of Canada may supply single copies of the dissertation.

BACK CLOSE WINDOW

A Fleet-Based Machine Learning Approach for Automatic Detection of Deviations between Measurements and Reality

Conference Proceedings: 2019 IEEE Intelligent Vehicles Symposium (IV)

Author: Jakob Pfeiffer; Peter Wolf; Roberto Pereira

Publisher: IEEE

Date: 9-12 June 2019

Thesis / Dissertation Reuse

The IEEE does not require individuals working on a thesis to obtain a formal reuse license, however, you may print out this statement to be used as a permission grant:

Requirements to be followed when using any portion (e.g., figure, graph, table, or textual material) of an IEEE copyrighted paper in a thesis:

1) In the case of textual material (e.g., using short quotes or referring to the work within these papers) users must give full credit to the original source (author, paper, publication) followed by the IEEE copyright line © 2011 IEEE.
2) In the case of illustrations or tabular material, we require that the copyright line © [Year of original publication] IEEE appear prominently with each reprinted figure and/or table.
3) If a substantial portion of the original paper is to be used, and if you are not the senior author, also obtain the senior author's approval.

Requirements to be followed when using an entire IEEE copyrighted paper in a thesis:

1) The following IEEE copyright/ credit notice should be placed prominently in the references: © [year of original publication] IEEE. Reprinted, with permission, from [author names, paper title, IEEE publication title, and month/year of publication]
2) Only the accepted version of an IEEE copyrighted paper can be used when posting the paper or your thesis on-line.
3) In placing the thesis on the author's university website, please display the following message in a prominent place on the website: In reference to IEEE copyrighted material which is used with permission in this thesis, the IEEE does not endorse any of [university/educational entity's name goes here]'s products or services. Internal or personal use of this material is permitted. If interested in reprinting/republishing IEEE copyrighted material for advertising or promotional purposes or for creating new collective works for resale or redistribution, please go to http://www.ieee.org/publications_standards/publications/rights/rights_link.html to learn how to obtain a License from RightsLink.

If applicable, University Microfilms and/or ProQuest Library, or the Archives of Canada may supply single copies of the dissertation.

BACK CLOSE WINDOW