Mitigating Bias in Bayesian Optimized Data while Designing MacPherson Suspension Architecture

Sinnu Susan Thomas, *Member, IEEE*, Guillaume Lamine, Jacopo Palandri, Mohsen Lakehal-ayat, Punarjay Chakravarty, Friedrich Wolf-Monheim, and Matthew B. Blaschko

***Abstract*—With the rise of artificial intelligence, the automotive industry searched for novel ways to improve future product design. We focus on designing automatic MacPherson suspension architecture for the automotive sector. It takes time for an automotive engineer to design vehicle parts and thus slows the pace of innovation in this field. Given the car's particular kinematic characteristics, we propose to predict an architecture by positioning the hardpoints. This work deals with the biased data generated using the discipline models using the dataset shift learning paradigm. The optimized data are created with random and uniform sampling, with more samples with random sampling. We resolve the bias in the data, using a novel criterion for tuning the kernel mean matching and a weight estimation algorithm and designing the required target characteristics.**

***Impact Statement*—In the early stages of vehicle suspension design and development, decisions are generally made regarding suspension architecture within the constraints of a given packaging space, vehicle weight, and required travel. Kinematic characteristics are traditionally derived from a disciplined model. A disciplined model is a software program that studies the behavior of interconnected rigid and flexible mechanical components as they undergo translational and rotational displacements resulting from applied forces or motion as measured by displacement, velocity, and acceleration. We design the suspension based on the data produced using the disciplined models. This paper helps the community of automotive engineers to use artificial intelligence and make their design process fast.**

***Index Terms*—MacPherson Suspension Architecture, Bias, Kernel Mean Matching, Covariate Shift, Bayesian Optimization.**

I. Introduction

THE possibilities of artificial intelligence have been explored extensively in the automotive industry during the past few years. The automobile manufacturing sector is interested in the automatic design process of the suspension architecture of their cars and an algorithm for suggesting suitable geometries for a given set of desired characteristics. In the manual process, we explore the design of the MacPherson suspension geometry of a car using the following design process:

- a specific set of characteristics needed for the car;
- designer chooses geometry;
- a model of the car is created and simulated in Multibody System (MBS) Software [1] or using Finite Elements Method (FEM) Model [2].
- designer chooses an updated geometry, and the overall procedure is repeated if the current characteristics do not match the desired ones.

The guarantee of desired characteristics of the car is assured by the presence of a human and his sufficient knowledge while choosing the geometry. While designing the geometry in an automotive manner, these aspects of the iterative process are mitigated. In this work, the focus is on the simplified problem: the positioning of the fixation points for the suspension of the rear wheels of a car. These fixation points are also called hardpoints (HP). The positioning of HP for MacPherson suspension [3], [4] of the rear wheels in a car is generated using Adams [5] - the MBS software resulting in the kinematics performances in terms of target characteristics of the car. The forward problem is to design target characteristics from the hardpoints. However, since we have the desired target characteristics, we pose the design as an inverse problem in the proposed approach.

The data generated using the discipline models were uniformly random sampled, and those using Bayesian Optimization (BO) [6, 7]. The algorithm's performance greatly depends on the data distributed in the search space. The data created using discipline models are costly and under different sampling techniques. The data we created had a bias in the sampling. Even though the data is biased, these data give details about different target characteristics. We conduct a broad study on the influence of the data distribution on the performances of a generated data set. Generally, we assume the training and the testing data are identically distributed. We focus on a more general learning approach depending on the data distribution. We use Dataset Shift and Biased Learning for two initial training sets. The two training sets are created using random and uniform sampling of BO experiments.

To compare two training sets using Dataset Shift, we use Kernel Mean Matching (KMM) for a weighted sample of the data set to optimize the performance of the generated test set. We introduce a novel criterion for optimizing the parameters of the KMM in this paper.

Finally, we study a general learning approach for different

"FORD supported the work through the FORD-KU Leuven Alliance. This research received funding from the Flemish Government under the "Onderzoeksprogramma Artificiële Intelligentie (AI) Vlaanderen" programme."

Sinnu S. Thomas is with the School of Computer Science and Engineering, Digital University Kerala 695317 India. e-mail: sinnu.thomas@duk.ac.in

G. Lamine is with Eura Nova, Belgium. e-mail: guillaume-lamine@hotmail.com

J. Palandri, M. Lakehal-ayat, and F. Wolf-Monheim are with Ford Research & Innovation Center, Süsterfeldstraße 200, 52072 Aachen, Germany. e-mail: {jpalandr, mlakehal, fwolf5}@ford.com

P. Chakravarty is with Planet, USA. e-mail: punarjay@gmail.com

Matthew B. Blaschko is with the Department of Electrical Engineering, KU Leuven, 3001 Belgium. e-mail: matthew.blaschko@esat.kuleuven.be

Manuscript received October 12, 2022; revised February 7, 2023, and March 27, 2023; accepted May 04, 2023.

training sets to see the performances. We discuss some insights on the influence of the initial training data distribution on the final regression performances. Lastly, we find an algorithm for a dataset with a particular training mode, best for an automotive design.

The readers are encouraged to read our previous work [6, 7] to get a broader perspective on the design of MacPherson architecture. The current paper aims to reuse the data generated by our previous work [6, 7] using BO to train a learning algorithm that could provide exciting answers to the regression problem. If the dataset gives enough insight into the entire search space, the estimation function is as good as the BO but with no time delay.

The main contributions of this paper are as follows:

- Designed a Data Shift model for the automotive data having a bias.
- Designed a novel criterion for optimizing the parameters of the KMM. KMM is tuned manually, but in the proposed approach, KMM is tuned according to the available data.
- Verified the validity of the approach for the toy problem and several suspension design settings. Several suspension design settings indicate several hardpoints of the suspension. The details are given in the Experiments Section.

This paper is organized into four sections. Section 2 gives a brief overview of the state-of-the-art approaches in this area. Section 3 proposes an optimal design for the suspension. Simulation results are presented in Section 4. Lastly, the final section concludes the paper.

II. Literature Review

Engineers have been designing the mechanical architecture for decades before using machine learning and data for optimization. The automatic design community has developed tools to find the best characteristics of the design of cars depending on their suspension architectures. The MacPherson suspension architecture is a commonly used architecture for vehicles. The positioning of the hardpoints alters the static and kinematic performances of the car. The state-of-the-art approaches [8]–[12] use mechanical models that perform a space search with the help of human experts.

Various statistical solutions exist in the literature, and better algorithms to optimize new mechanical designs are at ease with the growing hardware computing power and artificial intelligence. In many scenarios, the classical gradient-based optimization algorithms [13] is used. However, because the search spaces are usually vast and no derivatives are available, genetic algorithms (GA) are often utilized [14, 15].

There are many approaches where generative models are used [16]–[18] and are used when the search space is vast, data is not condensed in tabular arrays, and the computation of each point in the space is costly. The design also uses descriptive models in the literature [6, 7]. GA needs to train the same model on multiple hyperparameters, whereas BO is used to train a single model, thus making it computationally efficient. BO is derivative-free global optimization [19]–[21] used for expensive functions such as computing ADAMS. The training of neural networks is computationally expensive, whereas optimization using BO is computationally efficient.

BO is the optimization procedure, which uses a surrogate model approximating the truly optimized setting [19, 20, 22]–[25]. These surrogate models can be Artificial Neural Networks (ANN), Gaussian Processes (GP), and most classical descriptive models. BO has been studied in many fields for the optimization of unknown functions. The performance of BO is highly dependent on the choice of acquisition function made to find the best-observed value. Nguyen et al. [26] proposed convergence criteria for these acquisition functions to avoid unwanted evaluations. It has been applied with great success in machine learning applications [21], analog circuits [27], voltage failures [28], aerospace engineering [29], asset management [30], pharmaceutical products [31], laboratory gas-liquid separator [32], multi-objective optimization [33], electron lasers [34], controllers, road network design [35, 36] and autonomous systems.

We assume that the training and test datasets are, i.i.d. drawn from a standard probability distribution. A dataset shift occurs when the distribution on either side does not match. Covariate shift [37] is a standard data set shift caused by sample selection bias. This problem is solved by reweighting the training points such that the means of the training and test points in a reproducing kernel Hilbert space (RKHS) are close. Kernel Mean Matching (KMM) [38] is a milestone in density ratio or Radon - Nikodym derivative estimation problem. It gives density ratios between training and test data by minimizing their maximum mean discrepancy (MMD) in a RKHS [39]. MMD [40] witness the difference in distributions. It converges quickly and is not specific to any distribution [40]–[44]. The convergence rates of KMM depend on some regularity measure of the regression function and the capacity measure of the kernel [45]. The standard KMM is tuned using MMD to estimate the essential weights. However, the parameters are difficult to adjust. Hence Miao et al. [46] used Normalized Mean Squared Error to tune the parameters of KMM. Wasserstein distance (WD) is very efficient when we want to move from one optimal design to another. In this paper, we use WD to tune the KMM to transport it from one suspension design to another.

We limit this work to descriptive models as the designed problem contains tabular data, and the amount of data is reduced but, in practice, is relatively inexpensive to achieve. A point of a search space from ADAMS takes about 10 minutes of computation. The use of descriptive models to learn the relations between mechanical architecture and its characteristics exist in the literature. However, to our knowledge, the reuse of data generated in a biased way for mechanical design was never studied.

III. Data Shift in Automotive Setting

In this approach, we use the data generated using BO and this technique for tuning optimization of the Kernel Mean Matching algorithm useful for dataset Shift.

A. Inverse Problem in BO

BO optimizes black-box function f with large search spaces and expensive models. The process uses information from the problem evaluated at a certain point of the search space without any gradient information. BO adopts a search strategy depending on the statistical criteria to maximize and build a probabilistic surrogate model of the black box. Acquisition functions such as the Maximum Probability of Improvement (MPI), the Expected Improvement (EI), and the Confidence Bounds (CB) [19] decide upon the point in the search space to be evaluated.

The surrogate model represents the approximated underlying probability distribution of the evaluated data points. A popular surrogate model is Gaussian Process (GP), and we chose GP for this approach since it is a non-parametric model and the structure of the model is not known as apriori. This model resembles the actual distribution by a normal distribution. This work focuses on the Expected Improvement acquisition, Gaussian Process [6, 7] and the ones used to optimize the KMM parameters.

For the design of the suspension, we denote the set of possible statistics of the kinematics curve or the desired vehicle characteristics as $\mathcal{X}$ where $\mathcal{X} \subseteq \mathbb{R}^d$. We address the design of suspension hardpoints $\mathcal{Y}$ in this paper where $\mathcal{Y} \subset \mathbb{R}^m$ is a bounded domain for a given target $\mathbf{x} \in \mathcal{X}$. Automotive engineers design target characteristics $\mathcal{X}$ from the suspension design parameters $\mathcal{Y}$ and $\mathbf{y} \in \mathcal{Y}$. The d and m are the target characteristics and hardpoints space dimensions, respectively.

Given $g : \mathcal{Y} \to \mathcal{X}$ and a target $\mathcal{X}$, find a value $\mathbf{y}$ such that $g(\mathbf{y}) \approx \mathbf{x}$, we pose this problem as an inverse problem. This problem is an ill-posed problem that can have multiple $\mathbf{y}$ satisfying $g(\mathbf{y}) = \mathbf{x}$ for a given $\mathbf{x}$ or none. The existence of the solution can be restored by solving for the minimum norm solution

$$g^{\dagger}(\mathbf{x}) \;:=\; \arg\min_{\mathbf{y}\in\mathcal{Y}} \underbrace{\|g(\mathbf{y}) - \mathbf{x}\|^2}_{=:f(\mathbf{y})}, \tag{1}$$

where $g^{\dagger}$ defines a generalized inverse of g.

The desired suspension design parameters are designed using BO where g is computed using the multibody dynamic software, and BO is used to minimize f for $\mathbf{y}$ [6, 7].

B. Minimizing Risk functional

The primary purpose of this paper is to learn the inverse of the discipline models, such as learning geometrical points from the target characteristics. We create the dataset [6] generated using BO is biased, and we want to analyze its performance. Given a user-defined actual hypothetical distribution of the target characteristics for the feasible cars, we assume that the target set is distributed in the $\mathbf{y}$ domain following a Gaussian and uniform distribution. We infer the true probabilistic relation given a collection of empirical data using the frequentist approach [47]. Data learning is based on the general statistical problem of minimizing the functional risk based on empirical data.

We learn a L- Lipschitz continuous function $f : \mathcal{X} \to \mathcal{Y}$ that maps from the input space $\mathcal{X}$ to the output space $\mathcal{Y}$ in a noise free domain. In order to quantify the performance of a function f, we consider learning with a discrete loss function $\mathcal{L}$ [48]

$$\mathcal{L} : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}_+ \tag{2}$$

.

The risk $\mathcal{R}$ associated with the hypothesis f is then defined as the expectation of the loss function [49]

$$\mathcal{R}(f(\mathbf{x})) = \int \mathcal{L}(f(\mathbf{x}), \mathbf{y})\, dP(\mathbf{x}, \mathbf{y}) \tag{3}$$

where $P(\mathbf{x}, \mathbf{y})$ is a distribution that governs the probability that $\mathbf{x}$ and $\mathbf{y}$ are jointly observed in a correctly labeled data sample. Since we do not have access to the real probability distribution $P(\mathbf{x}, \mathbf{y}) \;= P(\mathbf{y}|\mathbf{x})P(\mathbf{x})$, and only empirical data is available in the training set, it is needed to formulate a novel constructive criterion based on empirical data that could serve to minimize the true risk functional.

The goal is to minimize the risk $\mathcal{R}$ over a class of functions $\mathcal{F}$ and find an optimal function f^*

$$f^* \;:=\; \underset{f\in\mathcal{F}}{\operatorname{argmin}}\; \mathcal{R}(f) \tag{4}$$

Empirical risk minimization substitutes an approximation to $\mathcal{R}$ based on an empirical sum over losses incurred on the finite n training sample, using an i.i.d. sampling assumption [50]. The empirical risk becomes

$$\mathcal{R}(f) \;\approx\; \hat{\mathcal{R}}(f) := \frac{1}{n}\sum_{i=1}^{n} \|f(\mathbf{x}_i) - \mathbf{y}_i\|^2 \tag{5}$$

C. Dataset Shift Paradigm

Usually, in machine learning, training, and test sets are assumed to be the same. The dataset shift paradigm is noticed when the available dataset is not sampled like the actual probability distribution of interest. Bias in experimental design, non-reproducibility of testing conditions, and evolution of the problem with time can be real-world data problems. We apply this concept since there is a substantial difference in the distributions of the training set for the testing set.

We face covariate shifts in the dataset created. In this paper, the data available for the training has been generated by a BO approach [7] for an automotive setting. The data has been developed for one vector of target characteristics (TC) of a hypothetical car, and the algorithm searches for the optimal geometrical positions of the vehicle. There is also a dataset of vectors randomly sampled in a small range around the target characteristic point, constituting an entire distribution.

$$\mathbf{y} \sim P_{bo}(\mathbf{y}) \text{ or } \mathbf{y} \sim P_{unif}(\mathbf{y}) \tag{6}$$

where P_{bo} is the distribution using the outputs from the BO sampling and P_{unif} is the distribution using the combination of random sampling and BO sampling. We mitigate this problem by creating a dataset mixture of training and test instances and then using importance-weighted cross-validation [51].

D. Kernel Mean Matching

Generally, in machine learning applications, we assume that the training and test datasets are, i.i.d. drawn from a standard probability distribution $P(\mathbf{x})$. Still, in practice, we face sample selection bias.

In the regression estimation, we assume the pairs $Z = (X, Y)$ of the input set X and the output set Y. For the sake of the reasoning, let us define the training set as $Z_{tr} = (X_{tr}, Y_{tr})$ and the test set as $Z_{te} = (X_{te}, Y_{te})$. These are respectively drawn from the training and testing probability distributions $P_{tr}(\mathbf{x}, \mathbf{y})$ and $P_{te}(\mathbf{x}, \mathbf{y})$ respectively. The subscripts tr denote the training samples, and te denotes the testing samples. In the proposed approach, we consider that the input labels $\mathbf{x}$ are available from the actual distribution, i.e., the test set, also called unlabeled data [38].

The weights are directly inferred by distribution matching without solving a probability density estimation problem. Instead, the weights are computed such that the means of the training and test points in RKHS are near Kernel Mean Matching. When the RKHS is universal, the mathematical solution for the weights is the ratio $\frac{P_{te}(\mathbf{x},\mathbf{y})}{P_{tr}(\mathbf{x},\mathbf{y})}$. We compute the bound on the convergence of the empirical means. We make covariate shift assumptions for this work. The learning process with two different distributions $P_{te}(\mathbf{x}, \mathbf{y})$ and $P_{tr}(\mathbf{x}, \mathbf{y})$ is unsolvable if the two distribution are arbitrarily far apart. In practice, nothing can be inferred if there is no similarity between a previous experience (training) and the current case (test). A simplifying assumption is made - the two probability distributions differ only by the input label distribution $\mathbf{y}$. Therefore, $P_{te}(\mathbf{x}, \mathbf{y}) = P(\mathbf{y}|\mathbf{x})P_{te}(\mathbf{x})$ and $P_{tr}(\mathbf{x}, \mathbf{y}) = P(\mathbf{y}|\mathbf{x})P_{tr}(\mathbf{x})$. The conditional probabilities remain unchanged. The problem with learning is minimizing the real risk.

$$\mathcal{R}(f(\mathbf{x})) = \int \mathcal{L}(f(\mathbf{x}), \mathbf{y})\ dP_{te}(\mathbf{x}, \mathbf{y}) \tag{7}$$

The joint distribution is unknown. We have already seen that the empirical risk minimization induction principle that one could instead minimize $\mathcal{R}_{emp}$

$$\mathcal{R}_{emp}(f) = \frac{1}{n}\sum_{i=1}^{n} \|f(\mathbf{x}_i) - \mathbf{y}_i\|^2 \tag{8}$$

We have a different $P_{te}(\mathbf{x}, \mathbf{y})$ and $P_{tr}(\mathbf{x}, \mathbf{y})$. The risk is minimized for the distribution $P_{te}(\mathbf{x}, \mathbf{y})$ in Eq. 7. We define the weights

$$\beta(\mathbf{x}, \mathbf{y}) = \frac{P_{te}(\mathbf{x}, \mathbf{y})}{P_{tr}(\mathbf{x}, \mathbf{y})} \tag{9}$$

Substituting into Eq. 7, we have

$$\mathcal{R}(f(\mathbf{x})) = \int \mathcal{L}(f(\mathbf{x}), \mathbf{y})\ \beta(\mathbf{x}, \mathbf{y})\ dP_{tr}(\mathbf{x}, \mathbf{y}) \tag{10}$$

For Eq. 10 to exist reweighting coefficients $\beta(\mathbf{x}, \mathbf{y})$ should be properly defined, i.e. that the support of $P_{te}(\mathbf{x}, \mathbf{y})$ should be contained in the support of $P_{tr}(\mathbf{x}, y)$.

As the weight function $\beta(\mathbf{x}, \mathbf{y})$ is considered to be unknown in the proposed approach, it must be estimated from the observed data. Note that the fact that we are in the covariate shift case simplifies further $\beta(\mathbf{x}, \mathbf{y})$ to $\beta(\mathbf{x}) = \frac{P_{te}(\mathbf{x})}{P_{tr}(\mathbf{x})}$ and estimate weighted empirical risk $\mathcal{R}_{emp,w}$.

$$\beta(\mathbf{x}) = \frac{P_{te}(\mathbf{x})}{P_{tr}(\mathbf{x})} \tag{11}$$

$$\mathcal{R}_{emp,w}(f) = \frac{1}{n}\sum_{i=1}^{n} \beta_i\ \|f(\mathbf{x}_i) - \mathbf{y}_i\|^2 \tag{12}$$

Let $\lambda : \mathcal{X} \to \mathcal{F}$ be a map into the feature space. $\nu : \mathcal{H} \to \mathcal{F}$ is the mathematical expectation in the feature space.

$$\nu(P_{tr}) := \int \lambda(\mathbf{x})\ dP_{tr}(\mathbf{x}) \tag{13}$$

In conclusion, if we know the kernel mean of the real test set distribution on the input space $\mathbf{x}$: $P_{te}(\mathbf{x})$, we can infer a suitable β by solving the following minimization problem [38] called as Kernel Mean Matching (KMM) procedure.

$$\begin{aligned} \underset{\beta}{\text{minimize}} \quad & \left\| \int \lambda(\mathbf{y})\ dP_{te}(\mathbf{x}) - \int \beta(x)\ \lambda(\mathbf{x})\ dP_{tr}(\mathbf{x}) \right\| \\ \text{subject to} \quad & \int \beta(\mathbf{x})\ dP_{tr}(\mathbf{x}) = 1,\ \beta(\mathbf{x}) \geq 0 \\ & \beta(\mathbf{x}) \geq 0 \end{aligned}$$

Following certain conditions, one can show that the solution of this problem gives $P_{te}(\mathbf{x}) = \beta(\mathbf{x})\ P_{tr}(\mathbf{x})$. Note that in order to solve such a problem, $P_{te}(\mathbf{x})$ and $P_{tr}(\mathbf{x})$ must still be known. In practice, the distributions are unknown, and we have the input samples Y_{tr} and Y_{te} of respective size m_{tr} and m_{te}.

The empirical estimate of the $\beta(\mathbf{x})$ expectation is normally distributed. β is bounded by Q. ($\beta(\mathbf{x}) \in [0, Q]$). We study the deviation between the empirical means of $P_{te}(\mathbf{x})$ and $\beta(\mathbf{x})\ P_{tr}(\mathbf{x})$ in feature space, given $\beta(\mathbf{x})$ is chosen perfectly in the population sense. Both expectations are replaced by their sample mean approximation and optimized the parameters of the empirical KMM using [38, Equation (11)].

$$\left\| \frac{1}{m_{tr}} \sum_{i=1}^{m_{tr}} \beta(x_{tr,i})\lambda(\mathbf{x}_{tr,i}) - \frac{1}{m_{te}} \sum_{i=1}^{m_{te}} \lambda(\mathbf{x}_{te,i}) \right\| \tag{14}$$

Eq. 14 is a constrained quadratic empirical KMM optimization that can be solved using interior-point methods or any other successive optimization procedure subject to constraints on β [37]. Researchers [52] use several methods to tune the parameters of KMM. We use statistical Wasserstein distance (WD) between the target and weighted train distribution. To use WD, we apply weighted sampling on the training set with the optimized weights given by the KMM algorithm. We use first-order WD to compare the two CDFs. WD compares two probability distributions when one variable is derived from the other by small, non-uniform noise.

IV. Experimental Results

We discuss the results using the current setting for the suspension design. We gather the data generated by [7]. Moreover, clean it for processing, then explore and understand the datasets' characteristics. The working of the KMM is validated and compared to the WD efficiency for better tuning. Once the best-weighted dataset is generated, the procedure for training and comparing the different models is shown. The final goal is to analyze their behavior for data size and decide which performs best. Data was produced in several batches with the uniformly distributed points and the optimally sampled points using [7]. For each iteration of the BO, multiple uniform samples were generated. The exact proportion between uniform and BO samples is 16. The BO chose the best next point after 16 uniformly generated points at every iteration.

In the first case, we analyze the position of the outer tie rod in x, y, z coordinates, and we assume other hardpoints to be constant at this time. This can lead to multiple constant target characteristics in the output space. Moreover, some dimensions behave peculiarly as they display a discrete sampling with two possible values. This is peculiar because all variables should vary in the continuous domain. This observation is important and could lead to strange behavior in KMM. Because this discrete behavior is due to rounding errors and only a few points present, those dimensions were assumed to be constant. We have $n_{BO} = 511$ data points from BO and $n_{unif} = 8218$ from uniform sampling. For the higher dimension HP architecture, $n_{BO} = 142$ and $n_{unif} = 0$. Note that there are only a few BO samples and no uniform ones.

A. Toy Regression Data

We consider first a toy regression model with input and output in the $\mathcal{R}^1$ space. We consider

$$f(x) = x^3 - x + w \tag{15}$$

where $w \sim \mathcal{N}(0, \frac{\text{range}(y)}{100} = 0.0375)$ and $x \in [-1.5; 1.5]$ and $y \in [-1.875; 1.875]$ (range$(y) = 3.75$, range$(x) = 3.0$).

Eq. 15 represents the data generator's forward model. It is equivalent to the ADAMS model of the car, which is assumed to be a function. However, this function is invertible, which is impossible for ADAMS. Therefore, this first model is positioned in a more straightforward setup.

B. Automotive dataset

The second model is a partial analytical model of a MacPherson architecture developed by [53]. A simple model that is easy to dash is purely kinematic and does not need any Ordinary Differential Equation solver (ODE). All the Suspension Design Factors are kinematic and geometric measures. The main goal of this model is to arrive at a level of complexity similar to the ADAMS model.

A summary of all the output characteristics and behavior in the 1HP dataset is shown in Table I. The discrete dimensions are removed from learning to make optimization more straightforward. The ranges of each variable are stated but without absolute reference in the space for data privacy reasons. This lets us realize how much variable space is covered in mechanical units. All higher-dimension HP setup values vary in input-output domains, and no variables are removed.

The final dimension spaces are the following: for the one HP setup, there are the (x, y, z) dimensions in input space and only 19 characteristics dimensions in output space; and for the higher dimension HP setup, there are $3 \cdot 8 = 24$ dimensions in input space and all the 19 characteristics dimensions in output space. The quantitative values of the positions of the hardpoints and values for the target characteristics is normalized between $[0, 1]$ relative to the full range of each dimension.

Once the two models are defined, the data is generated similarly to the one in the ADAMS dataset. The data is generated for uniform sampling straightforwardly using the models. The samples using BO are developed for both models. The parameters for BO were chosen according to the original ADAMS experiment. The datasets from ADAMS are relatively scarce to be able to make a suitable study and extract accurate conclusions. To cope with this issue, a broader study is made on a toy regression and a sub-model of MacPherson to generalize the observations better.

C. Validation of KMM

We study novel criteria to tune the KMM algorithm and analyze how the underlying data distribution influences the performances of the KMM and the training. We validate the functioning of the KMM using the toy regression problem with a linear model. Moreover, because computing the WD in dimension problems is costly, we approximate it by the L_2 norm of the WD between each pair of matching dimensions.

We optimize the KMM algorithm for different criteria to find the best σ parameter. One approach is a pure grid search of σ space and choosing the best result. The parameter σ is the kernel size in kernel mean matching. The sigma space is taken between $[-1, 1]$. The other approach is a BO to find the best point. Both approaches were simulated for 30 iterations and compared using final histograms and evolution of criteria with size.

We tested our results on all the four datasets we have created, such as toy problem, simple Car model, 1 HP, and 2HP. Fig. 1 and 2 show the KMM parameter without tuning. We choose the σ parameter of the kernel purely in a heuristic manner. The kernel bandwidth is chosen as the median of the pairwise distances between samples [54], [55]. We tested for a customarily distributed target set. We choose the mean, variance, and covariance matrix of the target normal distribution, respectively, as the mean of the BO set, and the variance and covariance of the BO set divided by a factor of 5. The training set used for the example is the BO set of the toy regression. Once the target set and σ factor are chosen, the optimal weights are computed with the KMM. We see the histograms of the initial train and target set in Fig. 1(a) and the weighted sample of the BO set with the target set in Fig. 1(b).

#	Name	Description	Behaviour 1 HP
1	`castor_angle`	Castor angle	Constant
2	`castor_trail`	Castor trail	Constant
3	`castor_offset`	Castor offset	Discrete
4	`kingpin_angle`	Kingpin angle	Constant
5	`scrub_radius`	Scrub radius	Constant
6	`kingpin_offset`	Kingpin offset	Discrete
7	`EAL`	Effective Arm Length	Continuous
8	`RCH`	Roll center height	Continuous
9	`FLCA_L2W_Ratio`	Front Lower Control Arm Ratio	Constant
10	`Ackerman_20deg`	Ackerman at 20 deg	Continuous
11	`Ackerman_fulllock`	Ackerman in full lock	Continuous
12	`Bump_Steer`	Bump steer	Continuous
13	`Bump_Camber`	Bump Camber	Continuous
14	`Bump_Caster`	Bump Caster	Discrete
15	`Kin_WC_Rec`	Kinematic wheel centre recession	Continuous
16	`Damper_Ratio`	Damper ratio	Discrete
17	`Spring_Ratio`	Spring ratio	Discrete
18	`Roll_Steer`	Roll steer	Continuous
19	`Roll_Camber`	Roll camber	Continuous

TABLE I: Complete names of the output characteristics of the ADAMS model for both 1 HP and 8 HP. The behavior in the 1 HP setup is noted. All the constant and discrete dimensions were removed for learning. Discrete means generally that the data had two values very close to each other, only due to rounding errors. Moreover, there was a high bias where most points presented one of the two values. For more details about the meaning of these measures, see Thomas et al.([6], and [7])

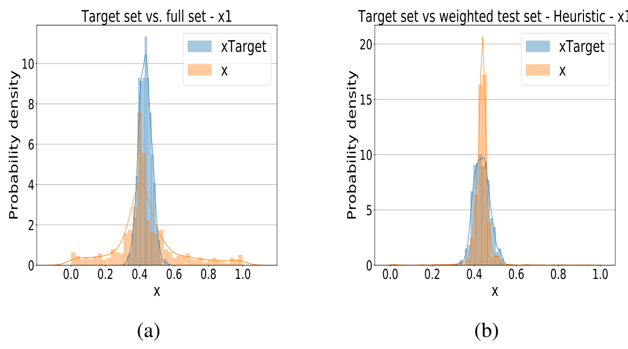


Fig. 1: (a) Unweighted training set. (b) Weighted training set (heuristic choice). Comparison of the histograms and estimated densities between the unweighted and weighted training sets with the weights chosen heuristically. Target sets are normally distributed.

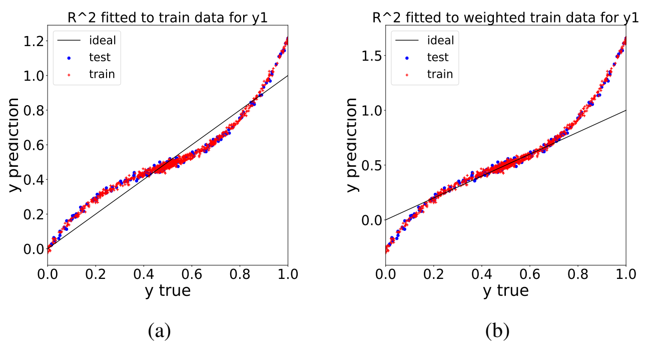


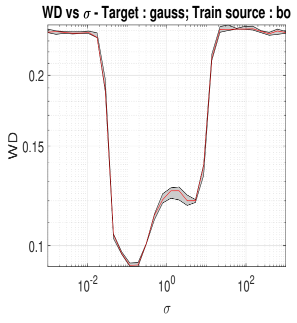
Fig. 2: (a) Unweighted training - $R_w^2 = 0.63$. (b) Weighted training (heuristic weights) - $R_w^2 = 0.89$. Comparison of the resulting Coefficient of determination plots between the linear model predictions for weighted and unweighted training. The y_{pred} predicted y are plotted against the real values y. The perfect situation is when the data points are aligned from 0 to 1. Target sets are normally distributed.

After computing the weights, a simple linear regression model is trained to observe the influence of the KMM. If the model fits the ground truth everywhere, the weighted training has less margin for improvement [56] as shown in Figs. 2(a) and 2(b).

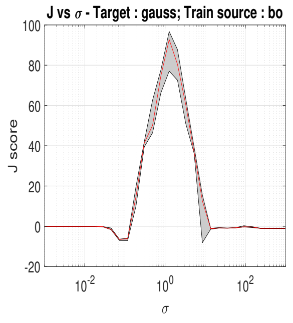
Fig. 1(b) shows that the two probability density functions, approximated with the histograms, are similar to Fig. 1(a). It is characterized by WD $= 0.012$ and J $= -1.11$. In Fig. 2(a) and 2(b), the linear model fits at the center of the space as the target distribution is concentrated at the center of the space. This is a biased estimation as it assumes that the weights computed for the test set are perfectly weighting the $P_{te}(\mathbf{x})$ to obtain the $P_{targ}(\mathbf{x})$. Moreover, it is also biased as the $P_{targ}(\mathbf{x})$ distribution is an abstract distribution that could even be impossible to exist in the search space.

Next, we see the evolution of the σ parameter for the KMM performance metrics such as WD, J score [57] and Maximum Mean Discrepancy (MMD), and time in Fig. 3. MMD [40] is a criterion that determines whether two sets of data are from the same or different probability distributions. The behavior of the different metrics stays quite similar for all the datasets, and the analyses here can be generalized to the other datasets studied. The performances were computed using cross-validation for each σ parameter. We use cross-validation to estimate the final metric for the whole set. This lets us know how much the metrics vary for the same σ, train, and target set. Fig. 4 shows the histogram of the target and weighted train sets, and the weights of the weighted training set are computed for four different σ parameters.

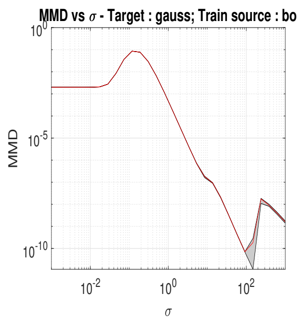
We see from Fig. 4(b) that the WD metric presents a good landscape for optimization, and there is one optimal point for σ that offers the smallest WD. This property is desirable for



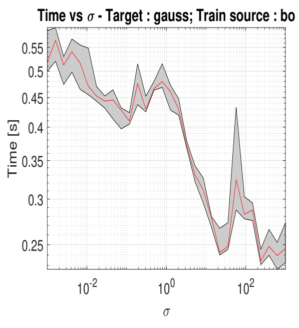
(a)



(b)

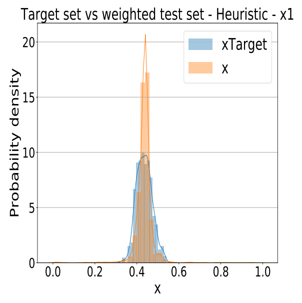


(c)

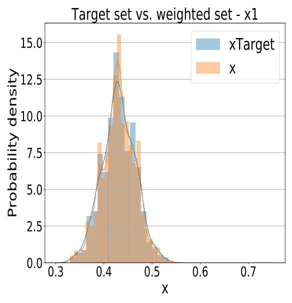


(d)

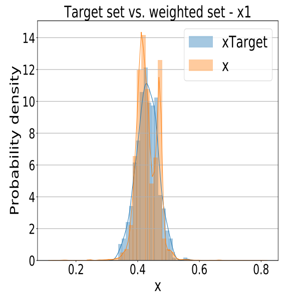
Fig. 3: (a) WD. (b) J. (c) MMD. (d) Time. Evolution of the median and $0.25-$, $0.75-$quantiles of the metrics WD, J, and MMD, and the time with the parameter σ. The values are obtained by cross-validation ($CV = 10$). Dataset : toy regression; Train distribution : BO; Target distribution : Normal.



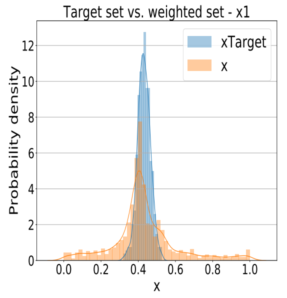
(a)



(b)



(c)



(d)

Fig. 4: (a) Heuristic choice - $\sigma = 0.19819$. (b) WD optimum - $\sigma = 0.0026$. (c) J optimum - $\sigma = 0.3039$. (d) MMD optimum - $\sigma = 0.9237$. Comparison of the optimization approaches on their best chosen σ parameter with histograms and estimated probability densities of the weighted training and target sets. Target sets are normally distributed.

BO to avoid getting stuck at a local minimum. At the extremes, when σ is very small or very big, the WD is maximal as the weighted training set, and the initial train set is the same. The weighted set is unchanged, and the weights converge to all. Moreover, the quantiles are narrow compared to the broad scale of variation of the median WD. We also see that the estimation of the distribution functions for the target set and the weighted training set are similar.

In Fig. 4(c), the spread is higher than in WD. It is about one order of difference as the ratio between inter quantile space and full median range is about $1e-3$ for WD and $1e-2$ for the J score. The J score presents, in a general way, more local minima, which was observed at different dataset sizes and different datasets. Fig. 4(c) shows the same distribution function for the target and weighted training sets.

Fig. 4(d) shows that MMD is not a criterion for higher-order optimization as it is biased towards higher σ. KMM converges quickly to all ones for the weights, and the time decreases with the increasing σ. We assume the optimal σ is found for the smallest WD. WD is a consistent metric for the full dataset size of 1000 points as it has a good histogram overlay and usually agrees with the J score.

Next, we saw the effect of the KMM performance metric w.r.t. on the dataset size. We evaluate how different metrics vary with the data size for an optimal σ parameter. The samples are produced with BO and a normal distribution for the target. Fig. 5 are produced with BO samples and a normal distribution for the target. The metrics are drawn with median and $0.25-$ $0.75-$ quantiles through cross-validation to avoid over-fitting.

From Fig. 5(d), we see that the combination BO + WD and GS + WD seem to agree most of the time concerning the range of σ. This is not the case between BO + J and GS + J as shown in Fig. 5(b) as the landscape of the WD metric is straightforward with one global minimum compared to the J score, the BO converges more often to the actual minimum. This is not the case for the J score. Still, what is surprising is that both algorithms optimizing for the J score are the ones with the minor J score due to the presence of the multiple local minima of the J landscape. The general observation is that the BO does not give better results than the more straightforward GS approach. Indeed, GS gives smaller WD and smaller J when optimizing as the search space is relatively small (unidimensional). BO could lead to good results if we tried to tune all the parameters of the KMM simultaneously.

It is visible from Fig. 5(a) and 5(b) that all the J scores of the five approaches are not very distinct and that the WD approaches have a fluctuating J. However, despite the fluctuations, the algorithms optimizing the WD are BO + WD and GS + WD. The MMD is minimized by the BO + J combination as it results in the high σ values. The heuristic approach results in a constant value of σ because the choice depends not on data size but only on the nature of the

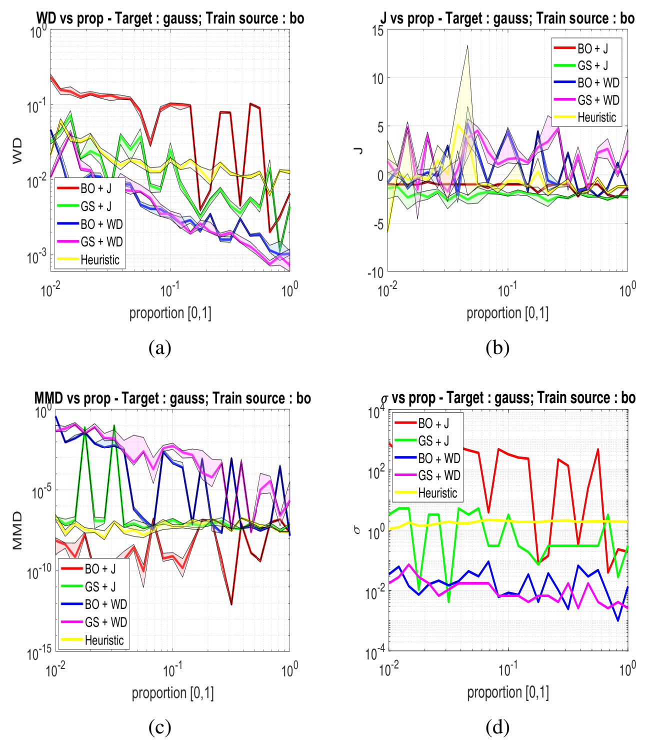


Fig. 5: (a) WD - Toy data. (b) J - Toy data. (c) MMD - Toy data. (d) σ parameter - Toy data. Evolution of the median and $0.25-$, $0.75-$quantiles of the optimization metrics WD and J as well as the MMD and the σ parameter with the data size and comparison between the five possible optimization combinations. The values are obtained by cross-validation ($CV = 10$). Dataset : toy regression; Full data size $n_{samp} = 1000$ (BO = Bayesian Optimization; GS = Grid Search)

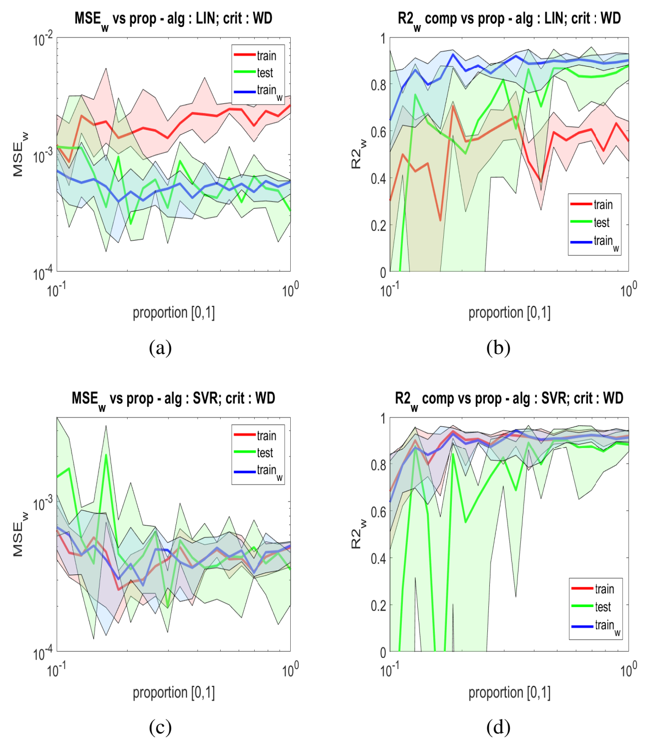


Fig. 6: (a) Toy data - Linear model - $n_{max,samp} = 1000$. (b)Toy data - Linear model - $n_{max,samp} = 1000$. (c) Toy data - Support Vector Regressor model - $n_{max,samp} = 1000$. (d) Toy data - Support Vector Regressor model - $n_{max,samp} = 1000$. Comparison of the medians and quantiles of the weighted MSE and R^2 between the two training modes: with and without weights. The values were produced by the SVR (Support Vector Regressor) and LIN (linear) models for the toy regression dataset - Test set values were computed on a test set with 20% of the full data. The weights were obtained with the WD criterion. The values are obtained by cross-validation ($CV = 10$). Full data size $n_{samp} = 1000$. ($train_w$ = weighted training; $train$ = unweighted training; $test$ = training direct on test set.)

distributions.

Finally, the algorithms have the same behavior except for unstable oscillations. Note that there is a general tendency for the WD to decrease with the data size. This is probably because the space is covered with more samples, so the weighted distribution can be refined and have a better statistical distance.

We now compare the different training modes: with and without the weights computed by the KMM. Fig. 6 and 7 shows the weighted Coefficient of Determination (R^2_w) and the weighted Mean Square Error (MSE$_w$). The values are computed with cross-validation (CV) to estimate the final scores on future points drawn with the median and the best-found CV result's $0.25-$, $0.75-$ quantiles. We compare two training modes and the performance on the test set when the machine is trained on the same test set. The weights are from the GS $+$ WD. For the training, the full set was split every time in train and test set in portions of 80% and 20%.

We compare the influence of the training mode (weighted or not) between the linear model (LIN) and the Support Vector Regressor (SVR) in Fig. 6, one can compare for the toy regression problem. The Linear model has its MSE$_w$ improved by the weighting; however, it is not statistically significant to conclude the SVR model as the model fits the ground truth properly, and weighting the points does not have a considerable effect. The test set result in green can be interpreted as the maximal possible performance as the algorithm is trained directly on the test set. However, as the test set is only 20% of the entire group, its performance is lower for the complete dataset's same total proportion p.

Fig. 7 shows the extreme situation in all datasets. Indeed, R^2_w is trending towards 1 with increasing dataset size for the datasets of toy and $adams1hp$. On the contrary, the datasets of $adams8hp$ and $simpCar$ are never above 0. Negative R^2 means that we are doing worse than just using the mean value of the output characteristic values; the model does not learn anything.

Two possible reasons for such behavior could be that we are missing data. The system converges to higher solutions with more data, or the function is under-specified. The model can not approximate the data by a function in this condition. This happens if the forward function used is not invertible. In our case, the inverse of the ADAMS model for $adams8hp$

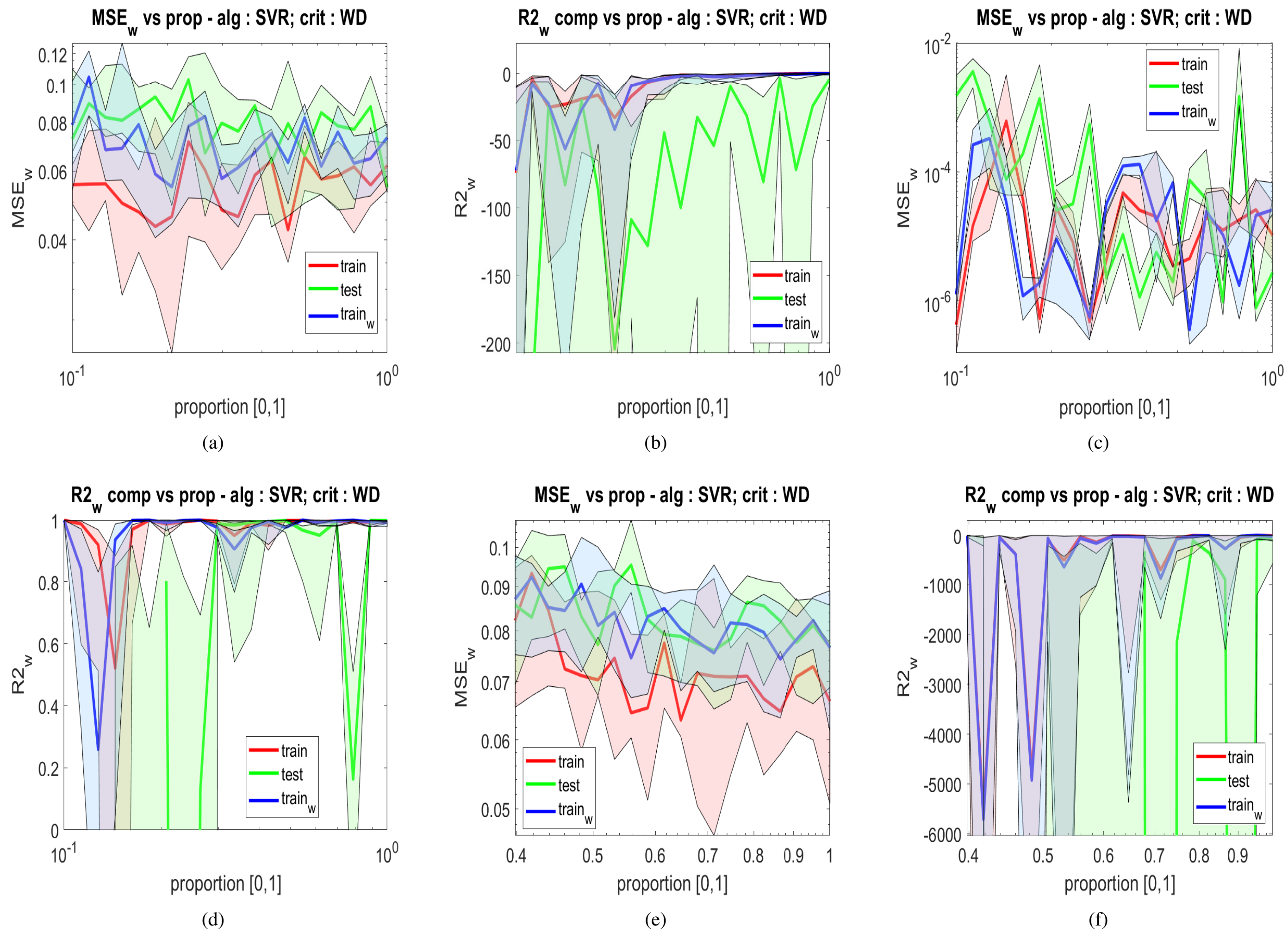
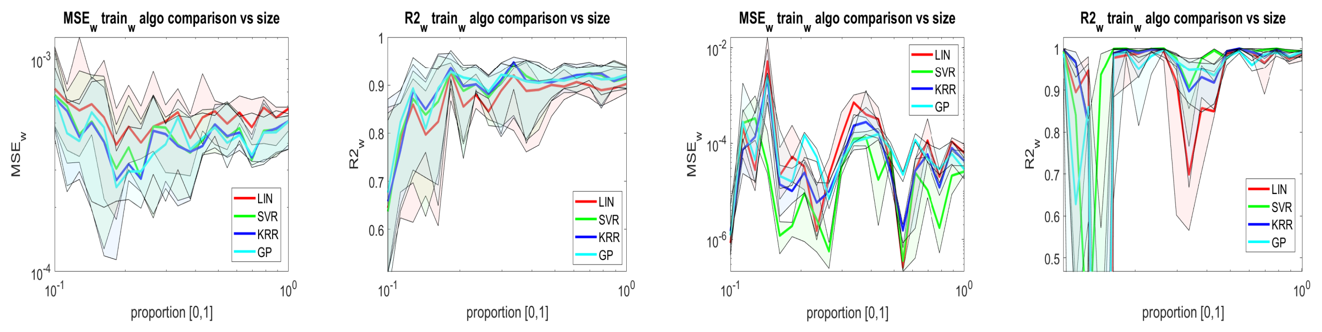


Fig. 7: (a) Analytical car model data (simpCar) - $n_{max,samp} = 1000$. (b)Analytical car model data (simpCar) - $n_{max,samp} = 1000$. (c) 1 HP ADAMS data - $n_{max,samp} = 511$. (d) 1 HP ADAMS data - $n_{max,samp} = 511$. (e) 8 HP ADAMS data - $n_{max,samp} = 142$. (f) 8 HP ADAMS data - $n_{max,samp} = 142$. Comparison of the weighted MSE and R^2 between the two training modes: with and without weights. The SVR model produced the values for the different datasets (simpCar, adams1hp, adams8hp) - Test set values were computed on a test set with 20% of the full data. The weights were obtained with the WD criterion. ($train_w$ = weighted training; $train$ = unweighted training; $test$ = training direct on test set.)

was not guaranteed. It is probably not invertible, and the $simpCar$ model is similar. The $adams1hp$ model, on the other hand, is invertible in the region studied. Indeed, if one trains a system in a forward way for the $simpCar$ model, the system has enough points to approximate the function, and the R^2 values are close to 1. We can probably accept that the non-invertibility of $simpCar$ and $adams8hp$ is the reason. Moreover, one could argue that the small number of points ($n_{samp} = 142$) for $adams8hp$ could be the reason. However, because $adams8hp$ is a black box model similar to $simpCar$, we assume that more data for the 8 HP setup does not increase the performance.

Remark also that the function is very well approximated and possibly simple because R_w^2 is almost 1 for the $adams1hp$ dataset. This is confirmed by comparing with the linear model for which $R_w^2 = 0.9869$ for the total dataset size of 1000 points. Because the $adams1hp$ has been sampled on a minimal space area (1 $[cm^3]$), the relations are well approximated by a linear hyper-plane. Because of these extreme situations for learning, more needs to be said about the efficiency of the training mode. The weighted MSE is worse for the models that are not invertible, but it needs to be more apparent for the 1 HP, and toy regression results were insignificant. This is still due to a proper function well approximated without weighting because of its non-complexity.

Next, we compare the performance metrics for the different systems (GP, SVR, LIN, and KRR) and their evolution with data size. We use the data of BO samples and the average target distribution to compare the influence of the data source on the final performances. Fig. 8(a) and 8(b) shows that GP, SVR, and KRR are better than the LIN model, even after the weighting. Fig. 8(c) and 8(d) shows that for 1 HP set, the SVR seems to be consistently better than the other systems.



(a) (b) (c) (d)

Fig. 8: (a) Toy data. (b) Toy data. (c) 1 HP ADAMS data. (d) 1 HP ADAMS data. Comparison of the weighted MSE and R^2 between the four trained algorithms for $a-b)$ the toy regression $d-e)$ 1 HP ADAMS data. The weighted trained models produce the weighted metrics. (LIN = linear; SVR = Support Vector Regressor; KRR = Kernel Ridge Regression; GP = Gaussian Process)

D. *Comparison of performances due to data source*

Next, we study the influence of the initial dataset distribution (BO-based or uniform) on the final performances of the test sets. The comparison is made for all systems and both target distributions: the standard and uniform. We compare based on MSE$_w$ with weights and data originating from the BO MSE$_{w,bo}$ and Uniform sets of samples MSE$_{w,unif}$ for each regression and are linked to the exact risk minimization as they have the same target. However, it can happen only if the KMM ideally chooses the weights.

Note that because there is no standard unbiased metric to compare all the combinations, the only analysis possible is to look at both MSE$_w$ values. If they agree on a data source with minor errors for both of them for a given target, then we say it produces better performances for the given target.

The MSE measures are produced for the complete datasets of size $n_{samp} = 1000$. The best system was selected for each algorithm for each target set and each training source using GridSearch. After that, each best system was evaluated on the test set. The whole experiment was repeated five times to estimate each combination's mean and standard deviation. The process was only done for the 1 HP and toy regression setups because they are the only ones where the system converges, as shown in Tables II and III.

Table II shows that for the toy regression, the linear model has both metrics with means smaller for the BO training data. It is the case for both targets. For the KRR and SVR linear model and the uniform target, the standards of the MSE$_w$ agree that the consistent training data for the constant target is the best. This is probably because the KMM works better to reproduce the suitable target distribution if the training distribution is non-zero at any point of the target. In other words, because the training samples of BO are concentrated at the center of the space, the KMM reproduces a good target if the target is uniform in the space.

In all other combinations, the results seem to be mainly influenced by the bias: the best data source is chosen by MSE$_{w,bo}$ and MSE$_{w,unif}$ are respectively the BO and Uniform training sets. Nevertheless, note that it is hard to affirm that the differences in the data source are significant given the standard deviation values for all these results. If there is a slight advantage in the mean effect, it is not evident and considerable. As seen in previous sections, the only significant difference is the recurrent fact that the SVR, GP, and KRR are close to each other but significantly better than the linear model. Ain these results are surprising.

Algo	TargSo	TrSo	MSE$_{w,bo}$	MSE$_{w,unif}$
LIN	normal	**bo**	(**5.38** $\pm$ **0.93**) e−4	(**4.44** $\pm$ **0.85**) e−4
		unif	(5.57 $\pm$ 0.36) e−4	(4.61 $\pm$ 0.63) e−4
	unif	**bo**	(**4.7** $\pm$ **0.46**) e−3	(**5.4** $\pm$ **0.59**) e−3
		unif	(5.3 $\pm$ 0.57) e−3	(5.5 $\pm$ 0.41) e−3
KRR	normal	bo	(**4.21** $\pm$ **0.68**) e−4	(3.75 $\pm$ 0.75) e−4
		unif	(4.61 $\pm$ 0.73) e−4	(**3.2** $\pm$ **0.25**) e−4
	unif	bo	(1.49 $\pm$ 0.25) e−4	(1.4 $\pm$ 0.23) e−4
		unif	(**1.47** $\pm$ **0.5**) e−4	(**1.15** $\pm$ **0.18**) e−4
SVR	normal	bo	(**4.27** $\pm$ **0.64**) e−4	(3.87 $\pm$ 0.69) e−4
		unif	(4.74 $\pm$ 0.67) e−4	(**3.26** $\pm$ **0.27**) e−4
	unif	bo	(1.64 $\pm$ 0.26) e−4	(1.53 $\pm$ 0.24) e−4
		unif	(**1.56** $\pm$ **0.55**) e−4	(**1.21** $\pm$ **0.13**) e−4
GP	normal	bo	(**4.33** $\pm$ **0.64**) e−4	(3.36 $\pm$ 0.87) e−4
		unif	(4.53 $\pm$ 0.74) e−4	(**3.11** $\pm$ **0.24**) e−4
	unif	bo	(**1.29** $\pm$ **0.27**) e−4	(1.33 $\pm$ 0.17) e−4
		unif	(1.46 $\pm$ 0.44) e−4	(**1.16** $\pm$ **0.20**) e−4

TABLE II: Artificial data - Toy regression - Weighted MSE values on the test sets for comparison between the training data distribution types (BO, Uniform) for each target data distribution type (Normal, Uniform). Values having the lowest mean are shown in bold. Both MSE$_{w,bo}$ and MSE$_{w,unif}$ mean the weighted MSE on the test data and weights coming respectively from BO and Uniform distributions

Concerning the 1 HP setup of ADAMS, the data source that seems to give the best results for most algorithms with a standard target is uniform training. The fact that it is recurrent for all algorithms seems quite significant. This is different from toy regression. Alternatively, the uniform target needs to be clarified. Finally, the linear model's performances are less far from the other linear model than the toy regression, showing the high linearity of the 1 HP problem.

Algo	TargSo	TrSo	MSE$_{w,bo}$	MSE$_{w,unif}$
LIN	normal	bo	(3.28 ± 4.19) e-4	(4.17 ± 3.76) e-5
		unif	$(\mathbf{2.85} \pm \mathbf{2.44})$ e-4	$(\mathbf{1.53} \pm \mathbf{0.50})$ e-6
	unif	bo	(1.05 ± 0.65) e-2	(3.91 ± 5.26)
		unif	$(\mathbf{7.1} \pm \mathbf{3.7})$ e-3	$(\mathbf{2.2} \pm \mathbf{2.6})$ e-3
KRR	normal	bo	(3.12 ± 4.11) e-4	(2.58 ± 2.03) e-5
		unif	$(\mathbf{2.84} \pm \mathbf{2.43})$ e-4	$(\mathbf{1.77} \pm \mathbf{0.68})$ e-6
	unif	bo	$(\mathbf{3.5} \pm \mathbf{1.3})$ e-3	(2.14 ± 1.05) e-2
		unif	(7.2 ± 3.7) e-3	$(\mathbf{3.1} \pm \mathbf{3.4})$ e-3
SVR	normal	bo	(3.13 ± 4.35) e-4	(1.51 ± 0.47) e-4
		unif	$(\mathbf{2.86} \pm \mathbf{2.42})$ e-4	$(\mathbf{2.44} \pm \mathbf{2.08})$ e-6
	unif	bo	$(\mathbf{4.3} \pm \mathbf{1.9})$ e-3	(5.63 ± 2.84) e-2
		unif	(7.0 ± 3.7) e-3	$(\mathbf{2.6} \pm \mathbf{3.6})$ e-3
GP	normal	bo	(3.05 ± 4.38) e-4	(9.93 ± 4.88) e-5
		unif	$(\mathbf{2.84} \pm \mathbf{2.44})$ e-4	$(\mathbf{2.44} \pm \mathbf{1.28})$ e-8
	unif	bo	$(\mathbf{5.3} \pm \mathbf{2.6})$ e-3	(6.48 ± 1.59) e-2
		unif	(7.1 ± 3.7) e-3	$(\mathbf{2.7} \pm \mathbf{3.7})$ e-3

TABLE III: ADAMS data - 1 HP - Weighted MSE values on the test sets for comparison between the training data distribution types (BO, Uniform) for each target data distribution type (Normal, Uniform). Values having the lowest mean are shown in bold. Both MSE$_{w,bo}$ and MSE$_{w,unif}$ mean the weighted MSE on the test data and weights coming respectively from BO and Uniform distributions

E. Benefits of KMM

The data collected using Bayesian Optimization for automotive design shows a selection bias in the data [58]. We assume the bias to be a covariate shift since it is impossible to correct shift absent assumptions [58]. We see that KMM is adaptive since it is not needed in advance to know the parameters of the regularization constant compared to other approaches. Since KMM uses the training and testing data in the learning phase, it converges faster, and the rate of several training samples n_{tr} increases. KMM converges at the rate of $\mathcal{O}(n_{tr}^{-\frac{1}{2}} + n_{te}^{-\frac{1}{2}})$ when the regression function lies in the RKHS [59]. KMM works very well in our high dimensional case since it estimates the important values at training points without going through density estimation [60, 61].

V. CONCLUSION

This paper deals with the dataset shift on the samples generated by the multibody simulation tool for automotive data. We analyzed the performance while tuning the Kernel Mean Matching using Wasserstein Distance and analyzed the regression performance for several parameters. This paper helps a car manufacturer decide the target characteristics based on specific hardpoint selection. In this work, we design the hardpoints of the vehicle using the required target characteristics. Many further studies can be conducted using a higher sample space and a higher number of hardpoints. The limitation of this method lies in the higher dimensional input space. The higher dimensional space leads to boundary problems in BO due to the curse of dimensionality.

REFERENCES

[1] M. Blundell and D. Harty, "The Multibody Systems Approach to Vehicle Dynamics 2nd Edition," 2014.

[2] R. J. Melosh, "Finite Element Analysis of Automobile Structures," *SAE Transactions*, pp. 1341–1355, 1974.

[3] E. S. MacPherson, "Vehicle Wheel Suspension System," 1953, US Patent 2,624,592.

[4] E. S. Macpherson, "Wheel Suspension for Motor Vehicles," 1953, US Patent 2,660,449.

[5] M. Software, "ADAMS: Multibody Dynamics Simulation Software," url: https://www.mscsoftware.com/product/adams, last checked on 2020-08-01.

[6] S. S. Thomas, J. Palandri, M. Lakehalayat, P. Chakravarty, F. Wolf-Monheim, and M. B. Blaschko, "Designing MacPherson Suspension Architectures using Bayesian Optimization," in *Benelearn*, 2019.

[7] S. S. Thomas, J. Palandri, M. Lakehal-ayat, P. Chakravarty, F. Wolf-Monheim, and M. B. Blaschko, "Kinematics Design of a MacPherson Suspension Architecture based on Bayesian Optimization," *IEEE Transactions on Cybernetics*, vol. 53, no. 4, pp. 2261–2274, Apr 2023.

[8] M. S. Fallah, R. Bhat, and W.-F. Xie, "H$^\infty$ Robust Control of Semi-active Macpherson Suspension System: New Applied Design," *Vehicle System Dynamics*, vol. 48, no. 3, pp. 339–360, 2010.

[9] A. A. Patil, "Mathematical Model for Kinematic Analysis of MacPherson Strut Suspension," SAE Technical Paper, Tech. Rep., 2016.

[10] S. Dehbari and J. Marzbanrad, "Kinematic and Dynamic Analysis for A New MacPherson Strut Suspension System," *Mechanics and Mechanical Engineering*, vol. 22, no. 4, pp. 1223–1238, 2018.

[11] Z. Chi, Y. He, and G. F. Naterer, "Design Optimization of Vehicle Suspensions with a Quarter-vehicle Model," *Transactions of the Canadian Society for Mechanical Engineering*, vol. 32, no. 2, pp. 297–312, 2008.

[12] X. Liu, M. Wang, X. Wang, C. Li, H. Guo, and J. Luo, "Hardpoint Correlation Analysis and Optimal Design for Front Suspension of A Formula SAE Car," *Australian Journal of Mechanical Engineering*, vol. 13, no. 2, pp. 67–76, 2015.

[13] M. L. Felzien and D. Cronin, "Steering Error Optimization of the MacPherson Strut Automotive Front Suspension," *Mechanism and Machine Theory*, vol. 20, no. 1, pp. 17–26, 1985.

[14] Q. Gao, J. Feng, and S. Zheng, "Optimization Design of the Key Parameters of MacPherson Suspension Systems using Generalized Multi-dimension Adaptive Learning Particle Swarm Optimization," *The Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering*, vol. 233, no. 13, pp. 3403–3423, 2019.

[15] Z. Su, F. Xu, L. Hua, H. Chen, K. Wu, and S. Zhang, "Design Optimization of Minivan MacPherson-strut Suspension System based on Weighting Combination Method and Neighborhood Cultivation Genetic Algorithm," *The Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering*, vol. 233, no. 3, pp. 650–660, 2019.

[16] B. Sauthoff and R. Lachmayer, "Generative Design Approach for Modelling of Large Design Spaces," in *World Conference on Mass Customization, Personalization, and Cocreation*, 2014, pp. 241–251.

[17] A. Nilsson and M. Thönners, "A Framework for Generative Product Design Powered by Deep Learning and Artificial Intelligence: Applied on Everyday Products," 2018.

[18] M. Cherti, "Deep Generative Neural Networks for Novelty Generation: A Foundational Framework, Metrics and Experiments," Ph.D. dissertation, 2018.

[19] E. Brochu, V. M. Cora, and N. De Freitas, "A Tutorial on Bayesian Optimization of Expensive Cost Functions, With Application to Active User Modeling and Hierarchical Reinforcement Learning," *CoRR*, vol. abs/1012.2599, 2010.

[20] B. Shahriari, K. Swersky, Z. Wang, R. P. Adams, and N. De Freitas, "Taking the Human out of the Loop: A Review of Bayesian Optimization," *Proceedings of the IEEE*, vol. 104, no. 1, pp. 148–175, 2015.

[21] J. Snoek, H. Larochelle, and R. P. Adams, "Practical Bayesian Optimization of Machine Learning Algorithms," in *NeurIPS*, 2012, pp. 2951–2959.

[22] J. Mockus, "On Bayesian Methods for Seeking the Extremum," in *IFIP Technical Conference*. Springer-Verlag, 1974, p. 400–404.

[23] M. Poloczek, J. Wang, and P. Frazier, "Multi-Information Source Optimization," in *NeurIPS*, vol. 30, 2017, pp. 4288–4298.

[24] S. F. Ghoreishi and M. Imani, "Bayesian Optimization for Efficient Design of Uncertain Coupled Multidisciplinary Systems," in *ACC*, 2020, pp. 3412–3418.

[25] R. Baptista and M. Poloczek, "Bayesian Optimization of Combinatorial Structures," in *ICML*, 2018, pp. 462–471.

[26] V. Nguyen, S. Gupta, S. Rana, C. Li, and S. Venkatesh, "Regret for Expected Improvement over the Best-Observed Value and Stopping Condition," in *ACML*, vol. 77, Nov 2017, pp. 279–294.

[27] W. Lyu, P. Xue, F. Yang, C. Yan, Z. Hong, X. Zeng, and D. Zhou, "An Efficient Bayesian Optimization Approach for Automated Optimization of Analog Circuits," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 65, no. 6, pp. 1954–1967, Jun 2018.

[28] H. Hu, P. Li, and J. Z. Huang, "Enabling High-Dimensional Bayesian Optimization for Efficient Failure Detection of Analog and Mixed-Signal Circuits," in *DAC*, 2019, pp. 17:1–17:6.
[29] R. Lam, M. Poloczek, P. Frazier, and K. E. Willcox, "Advances in Bayesian Optimization with Applications in Aerospace Engineering," in *AIAA Non-Deterministic Approaches Conference*, 2018, p. 1656.
[30] J. Gonzalvez, E. Lezmi, T. Roncalli, and J. Xu, "Financial Applications of Gaussian Processes and Bayesian Optimization," *CoRR*, vol. abs/1903.04841, 2019.
[31] S. Sano, T. Kadowaki, K. Tsuda, and S. Kimura, "Application of Bayesian Optimization for Pharmaceutical Product Development," *Journal of Pharmaceutical Innovation*, Mar 2019.
[32] J. Kocijan and A. Grancharova, *Application of Gaussian Processes to the Modelling and Control in Process Engineering*. Springer, 2014, pp. 155–190.
[33] H. Wang, H. Xu, Y. Yuan, J. Deng, and X. Sun, "Noisy Multiobjective Black-box Optimization Using Bayesian Optimization," in *GECCO*, 2019, pp. 239–240.
[34] J. Kirschner, M. Mutny, N. Hiller, R. Ischebeck, and A. Krause, "Adaptive and Safe Bayesian Optimization in High Dimensions via One-Dimensional Subspaces," in *ICML*, vol. 97, Jun 2019, pp. 3429–3438.
[35] J. Cui, B. Yang, B. Sun, and J. Liu, "Cost-aware graph generation: A deep bayesian optimization approach," *AAAI*, vol. 35, no. 8, pp. 7142–7150, May 2021.
[36] J. Cui, B. Yang, B. Sun, X. Hu, and J. Liu, "Scalable and Parallel Deep Bayesian Optimization on Attributed Graphs," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 33, no. 1, pp. 103–116, 2022.
[37] A. Gretton, A. Smola, J. Huang, M. Schmittfull, K. Borgwardt, and B. Schölkopf, *Dataset Shift in Machine Learning*. Cambridge, MA: MIT Press, 2008, pp. 131–160.
[38] J. Huang, A. J. Smola, A. Gretton, K. M. Borgwardt, and B. Schölkopf, "Correcting Sample Selection Bias by Unlabeled Data," in *NeurIPS*, 2006, pp. 601–608.
[39] W. Jitkrittum, P. Sangkloy, M. W. Gondal, A. Raj, J. Hays, and B. Schölkopf, "Kernel Mean Matching for Content Addressability of GANs," in *ICML*, vol. 97, 2019, pp. 3140–3151.
[40] A. Gretton, K. M. Borgwardt, M. J. Rasch, B. Schölkopf, and A. Smola, "A Kernel Two-Sample Test," *JMLR*, vol. 13, no. 25, pp. 723–773, 2012.
[41] Y. Chen, M. Welling, and A. Smola, "Super-Samples from Kernel Herding," in *UAI*, 2010, p. 109–116.
[42] F. Bach, S. Lacoste-Julien, and G. Obozinski, "On the Equivalence between Herding and Conditional Gradient Algorithms," in *ICML*, 2012, p. 1355–1362.
[43] S. Lacoste-Julien, F. Lindsten, and F. Bach, "Sequential Kernel Herding: Frank-Wolfe Optimization for Particle Filtering," in *AISTATS*, 2015, pp. 544–552.
[44] W. Y. Chen, L. W. Mackey, J. Gorham, F. Briol, and C. J. Oates, "Stein Points," in *ICML*, 2018, pp. 843–852.
[45] Y. Yu and C. Szepesvari, "Analysis of Kernel Mean Matching under Covariate Shift," *CoRR*, vol. abs/1206.4650, 2012.
[46] Y.-Q. Miao, A. K. Farahat, and M. S. Kamel, "Auto-Tuning Kernel Mean Matching," in *ICDM Workshops*, 2013, pp. 560–567.
[47] V. N. Vapnik, *Statistical Learning Theory*. Wiley-Interscience, 1998.
[48] J. Yu and M. Blaschko, "A Convex Surrogate Operator for General Non-Modular Loss Functions," in *AISTATS*, vol. 51, May 2016, pp. 1032–1041.
[49] M. Blaschko, "Advances in Empirical Risk Minimization for Image Analysis and Pattern Recognition," Ph.D. dissertation, ENS Cachan, Nov. 2014.
[50] V. N. Vapnik, *The Nature of Statistical Learning Theory*. Springer-Verlag New York, Inc., 1995.
[51] M. Sugiyama, M. Krauledat, and K.-R. Müller, "Covariate Shift Adaptation by Importance Weighted Cross Validation," *JMLR*, vol. 8, p. 985–1005, Dec. 2007.
[52] Y. Miao, A. K. Farahat, and M. S. Kamel, "Auto-Tuning Kernel Mean Matching," in *IEEE ICDM Workshops*, 2013, pp. 560–567.
[53] J. Gillberg and D. Zapardiel, "Steering Performance Dependence on Front Suspension Design," Master's thesis, Chalmers University of Technology, 2015.
[54] B. Schölkopf, A. J. Smola, F. Bach *et al.*, *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT press, 2002.
[55] M. Sugiyama, T. Suzuki, and T. Kanamori, *Density Ratio Estimation in Machine Learning*. Cambridge University Press, 2012.
[56] J. Quionero-Candela, M. Sugiyama, A. Schwaighofer, and N. D. Lawrence, *Dataset Shift in Machine Learning*. The MIT Press, 2009.
[57] W. J. Youden, "Index for Rating Diagnostic Tests," *Cancer*, vol. 3, no. 1, pp. 32–35, 1950.
[58] S. Rabanser, S. Günnemann, and Z. Lipton, "Failing Loudly: An Empirical Study of Methods for Detecting Dataset Shift," in *NeurIPS*, vol. 32, 2019.
[59] F. Li, H. Lam, and S. Prusty, "Robust Importance Weighting for Covariate Shift," in *AISTATS*, vol. 108, Aug 2020, pp. 352–362.
[60] T. Kanamori, S. Hido, and M. Sugiyama, "A Least-squares Approach to Direct Importance Estimation," *JMLR*, vol. 10, no. 48, pp. 1391–1445, 2009.
[61] M. Cheng and X. You, "Adaptive Matching of Kernel Means," in *ICPR*, 2020, pp. 2498–2505.

Sinnu Susan Thomas is currently a faculty member at the Department of Computer Science and Engineering, Digital University Kerala (formerly IIITMK), India.

Guillaume Lamine @ Eura Nova is currently a Machine Learning Engineer at EuraNova, Belgium.

Jacopo Palandri is currently a Project Manager and Research Engineer at the Ford Research and Innovation Center in Aachen (Germany).

Mohsen Lakehal-ayat is currently a Senior Engineer at the Ford Research and Innovation Center in Aachen (Germany).

Punarjay Chakravarty is currently a Staff Edge ML Engineer at Planet, USA.

Friedrich Wolf-Monheim is currently a Project Manager at the Ford Research and Innovation Center in Aachen (Germany) and a Lecturer at the Institute of Structural Mechanics and Lightweight Design of the RWTH Aachen University.

Matthew B. Blaschko is currently a faculty member at the Electrical Engineering Department, KU Leuven, Belgium.