



Data Fusion Techniques for Optimizing Motorsport Aeromap Generation: A Survey

Thomas James Henry Ongley Cranfield University

Tom-Robin Teschner Cranfield University, FEAS

Neil Ashton Amazon Web Services Inc

Efstathios Siampis Cranfield University, FEAS

Citation: Ongley, T.J.H., Teschner, T.-R., Ashton, N., and Siampis, E., "Data Fusion Techniques for Optimizing Motorsport Aeromap Generation: A Survey," SAE Technical Paper 2026-01-5058, 2026, doi:10.4271/2026-01-5058.

Received: 16 Oct 2025

Revised: 08 Jan 2026

Accepted: 16 Jun 2026

Abstract

This paper reviews data fusion strategies for generating aerodynamic databases and evaluates their suitability for motorsport aeromaps, with emphasis on the operational constraints specific to Formula One. A structured survey and classification of the state of the art is presented, grouping approaches into (i) surrogate-agnostic methods, (ii) kriging-based methods, and (iii) neural network-based methods. In addition, the survey explores advanced techniques currently underutilized in aerodynamic database applications but that show promise. These methodologies are discussed in the context of addressing limitations inherent in traditional approaches, such as dependency on nested sampling plans and linear correlation assumptions between low- and high-fidelity datasets. The review indicates that, although multi-fidelity data fusion is well established in

aerospace aerodynamic database generation, its direct translation to motorsport requires additional considerations. In the Formula One context, the most plausible deployment may involve fusing legacy and current datasets, rather than combining low- and high-fidelity evaluations of the same geometry. This shift in premise could increase exposure to negative transfer and therefore necessitate additional methods to minimize it. This study provides one of the first motorsport-focused reviews and syntheses of data fusion methods for aerodynamic database generation. It is intended to guide motorsport engineers and researchers toward more efficient and effective aeromap generation strategies. Collectively, the findings establish a foundation for subsequent phases of a broader project to minimize the number of data points required to generate an aeromap, with the present survey constituting the first part of that effort.

Keywords

Data fusion, Multi-fidelity, Transfer learning, Surrogate modeling, Kriging, Machine learning, Aeromaps,

Aerodynamic databases, Motorsport, Formula 1

Introduction

Motorsport engineers require an aerodynamic understanding of a vehicle across its entire range of operating conditions. Aeromaps are aerodynamic databases that capture key performance metrics, such as lift and drag coefficients, as a function of several operational parameters, such as front and rear ride

height, roll, steering angle, and yaw. By integrating these databases into vehicle models, engineers can fine-tune car setups and accurately pinpoint aerodynamic inefficiencies that need to be addressed, thereby driving the processes that underpin aerodynamic development. In Formula One (F1), teams are required to adhere to a set of sporting [1] and technical [2] regulations. Among the

former are the aerodynamic testing restrictions (ATR), which effectively limit wind tunnel testing time and computational fluid dynamics operations over a given period during the year. Consequently, there is a significant emphasis on optimizing the experimental and simulation methods to maximize development.

Though research on motorsport aeromaps is sparse, literature on aerodynamic databases from aerospace applications provides a useful proxy. In this aerospace context, surrogate models, also known as metamodels, emulators, or response surfaces, are employed to approximate the function of an underlying system based on a set of input and output data, known as training data or sample data. These models can then be used to quickly predict output values away from the training data points. The accuracy of these models depends on the quantity and placement of the samples as well as the type of surrogate model used. Optimized sample placement can improve modeling performance, which is achieved with static or adaptive sampling methods. The latter aims to position samples in unexplored and complex regions of the input space [3]. Another way of optimizing surrogate modeling is through data fusion. Data fusion works by leveraging a large set of inexpensive, low-fidelity data to predict the general trend of an underlying system and then correcting this trend with a smaller set of expensive, high-fidelity data. This multi-fidelity approach can yield better model performance than just using the high-fidelity data alone, thereby reducing the overall computational cost of the analysis. Data fusion is not limited to computer simulations, and it can also be applied to experimental cases, such as with wind tunnel data. Data fusion has been applied to aerodynamic databases successfully, with methods such as multi-fidelity kriging [4–6] and scaling functions [7–9] being the most common. Other, less common methods include neural network-based approaches [10, 11].

This paper provides a comprehensive review of data fusion techniques that have been applied to aerodynamic databases, filling a gap in the existing literature. The review aims to identify and discuss the strengths and limitations of current methods and proposes alternative methods from other surrogate modeling applications. The remainder of this paper is organized as follows: **Aeromaps** provides an overview of aeromaps. **Surrogate Modeling Methods** provides an in-depth discussion of surrogate modeling techniques found in the data fusion literature. **Data Fusion for Aerodynamic Databases** surveys data fusion approaches for aerodynamic databases, highlighting their strengths and limitations. Finally, **Discussion and Concluding Remarks** summarizes the key findings of this review and suggests potential directions for future research, including a brief review of data fusion methods that have been successfully applied in other applications.

Aeromaps

Aerodynamic design in motorsport was initially focused on minimizing drag by streamlining race car bodies to

maximize top speed. In the 1960s, engineers and designers began experimenting with cars fitted with wings. Unlike aircraft wings, these wings were intended to generate negative lift, known as downforce, with the aim of increasing tire force to improve cornering speed. The driver-engineer Jim Hall is usually credited with being the first person to implement downforce-generating wings on a race car with his Chaparral 2E [12]. Not long after, in 1968, F1 teams began introducing wings, and so began a new era of motorsport in which aerodynamic analysis became an integral part of race car design.

A motorsport aerodynamicist's high-level aim is to optimize the airflow around a car to maximize downforce per unit drag, to ensure a balanced front–rear aerodynamic load, and to maintain predictable and stable behavior across different operating conditions. This simplified view neglects other considerations, such as cooling optimization and drag reduction in high-speed straight-line conditions, but it helps to motivate the need for an aeromap. An aeromap is a motorsport-specific aerodynamic database, which describes the relationship between key aerodynamic outputs such as lift coefficient (C_L), drag coefficient (C_D), and aerodynamic balance (AB) as a function of several input parameters: front/rear ride heights, roll, steer, and yaw angles. These input and output parameters can be expanded to include more parameters depending on the desired fidelity. Aeromaps are integrated into vehicle simulations to evaluate the overall performance of the car, leading to an understanding of how the aerodynamics and setup can be optimized to reduce lap time. An example of a ride height aeromap generated in a wind tunnel by the Minardi F1 team and used in the work of Gadola et al. [13] is shown in [Figures 1 and 2](#). The contour plot corresponds to an unspecified aerodynamic coefficient (either drag, front downforce, or rear downforce). In [Figure 1](#), the black markers indicate the ride heights encountered during maximum acceleration and braking, effectively highlighting the extremes of the car's operational envelope. On the other hand, [Figure 2](#) shows the output ride heights from a lap time simulator, indicating the actual ride heights covered during a lap of a specific circuit.

The rest of this section will discuss how aeromaps are generated. First, the design of experiments (DoE), i.e., how the sampling of data is conducted, is discussed. Then, the different aerodynamic data generation methods within motorsport are discussed. This is followed by a brief overview of how surrogate modeling can be used to predict values away from the observed data. A summary of these steps, along with the associated methods, is shown in [Figure 3](#).

Design of Experiments

DoE is a systematic approach for planning experiments and simulations. In the context of an aeromap, this involves defining a set of operating condition values, or sample points, from which to measure the relevant output parameters (C_L , C_D , AB) to generate a training dataset for the surrogate model. The design of computer

FIGURE 1 Aeromap with maximum acceleration and braking ride heights.

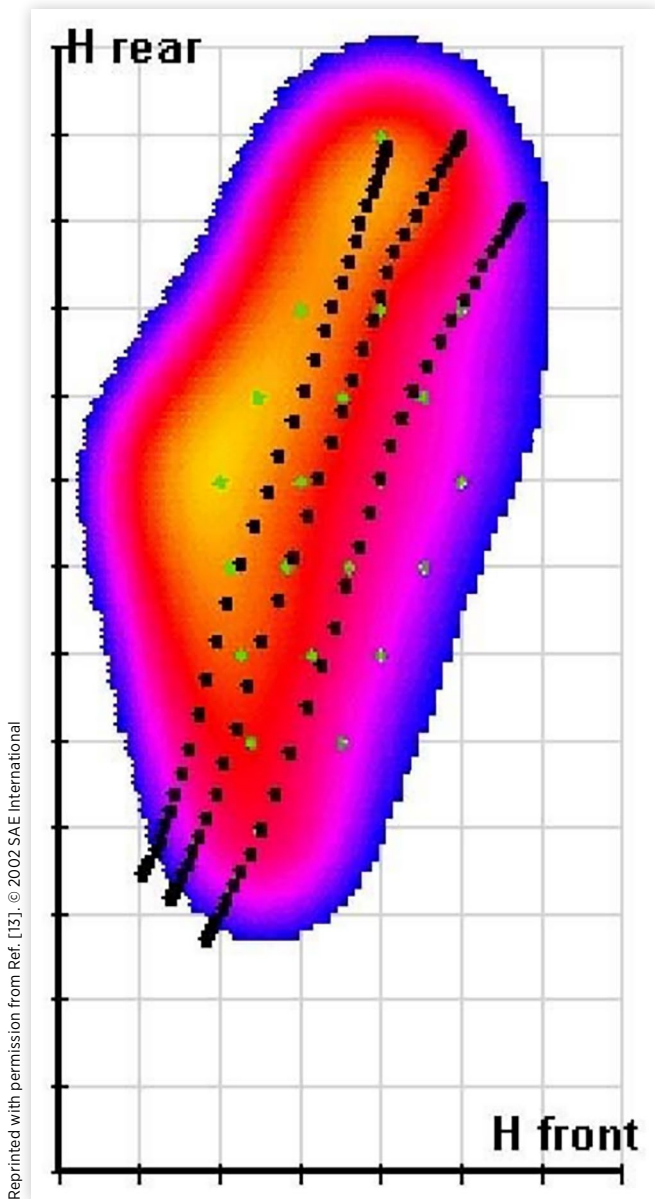
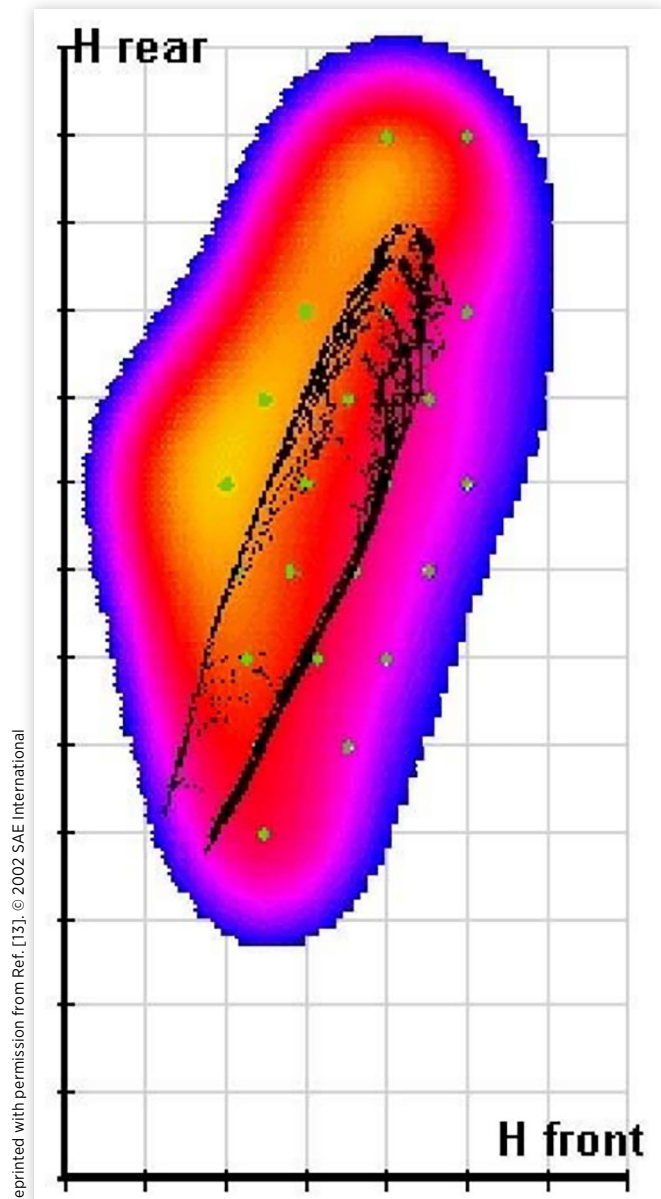


FIGURE 2 Aeromap with circuit lap time simulation ride heights.



experiments, also termed modern DoE (classical DoE refers to physical experiments), can broadly be classified into two groups: static and adaptive.

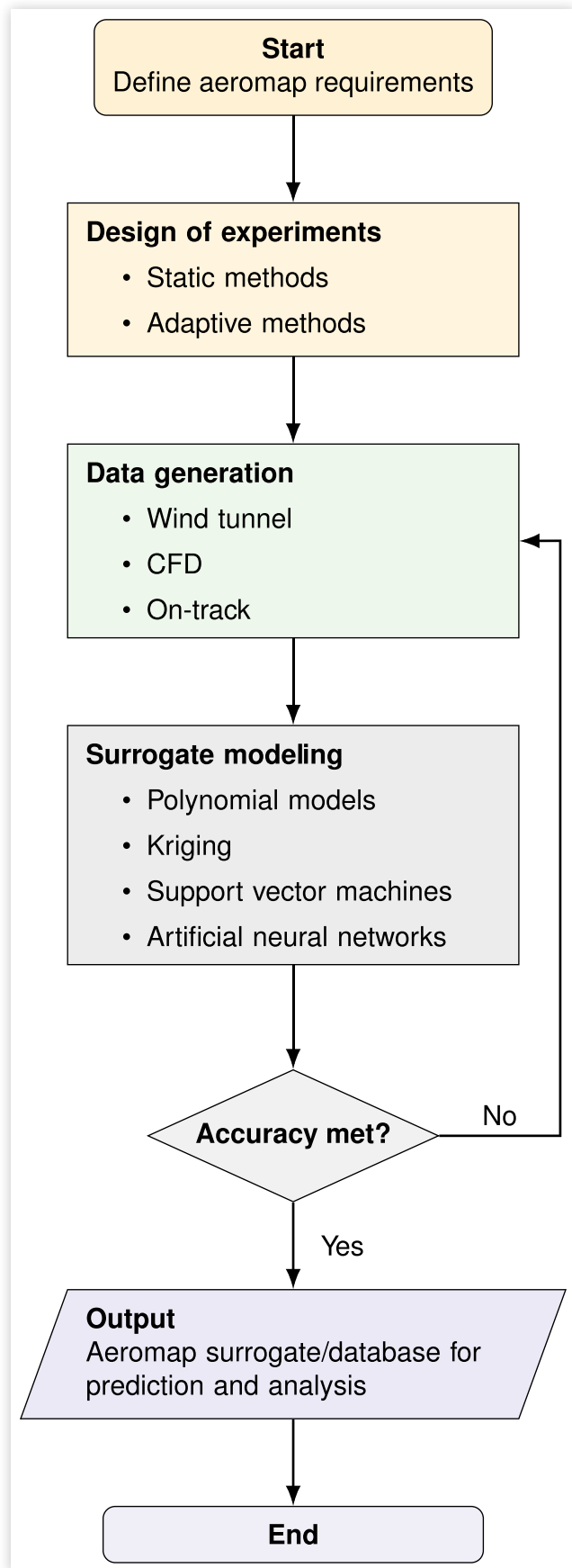
Static DoE refers to methods that generate a sampling plan before any data are generated. This approach is typically more straightforward to implement but may result in suboptimal sampling if the underlying function is not uniformly complex. Static methods include, but are not limited to, Latin hypercube sampling (LHS) [14], orthogonal array sampling (OAS) [15], Monte Carlo and quasi-Monte Carlo methods [3], and uniform sampling methods. With LHS and OAS being the most common within the engineering literature [3].

Adaptive sampling allows for a more efficient allocation of computational resources by focusing sampling efforts where they are most needed. This is achieved by

fitting a surrogate model to an initial sampling plan and using information from the surrogate to inform the placement of new samples. The two main components driving adaptive DoE are *exploration* and *exploitation*:

- Exploration involves the global exploration of the input domain to ensure that sample points are spread across the entire space. The goal is to prevent undersampling in any region, which could lead to imbalanced accuracy across regions of the model.
- Exploitation focuses on local regions where the underlying function is complex or exhibits high variability. By increasing the density of sample points in these areas, the surrogate model can capture complex areas of the data more effectively.

FIGURE 3 Workflow for aeromap generation, with the inclusion of an optional adaptive loop.



Adaptive DoE methods that use both exploration and exploitation to drive sample placement are known as *adaptive hybrid sampling* methods. Quantifying function complexity is achieved through methods such as cross-validation error, linearity approximations, and predicted error from Gaussian methods. The weighting given to these two components is also important and can have a significant effect on sampling performance [16]. A detailed discussion of static and adaptive DoE methods is beyond the scope of this review. For a comprehensive literature survey of both static and adaptive methods, readers are referred to Garud et al. [3].

Data Generation

This section introduces the three main ways of generating aerodynamic data in motorsport: wind tunnels, CFD, and on-track testing. This subsection provides brief context on aeromap data acquisition; detailed discussion of testing facilities, CFD methodology, or software tools is beyond the scope of this review.

Wind Tunnels Wind tunnels enable the generation of aerodynamic data through the use of a test geometry, which is typically a scale model in both motorsport and aerospace. Once the wind tunnel is operating, the model is rotated and translated via actuators across the pre-defined test points and a balance is used to measure the desired forces and moments. One of the unique aspects of a wind tunnel used for automotive or motorsport purposes is the presence of a moving belt under the model, also known as a moving ground plane, and a suction scraper, which is positioned ahead of the belt. These components prevent the creation of a ground plane boundary layer that would otherwise alter the flow field around the model. The moving ground plane is also important for accurately simulating the flow around rotating wheels, particularly in the region close to the ground.

Using a wind tunnel introduces some factors that are not present in an on-track scenario, such as walls, arms, and Reynolds number effects. Care must be taken to ensure that the presence of these effects has been properly accounted for when processing the data. In addition, simulating the curved flow a car experiences when cornering is not typically practical, and in the case of F1, attempting to create curved flow conditions is not permitted [1]. This restriction, along with additional constraints on model size and wind tunnel air velocity, means that the data obtained from F1 wind tunnels can deviate significantly from that found from on-track testing. For further information, Toet [17] provided an overview of the considerations required when performing wind tunnel testing in F1, and Bushnell [18] detailed a more general review of the scaling issues encountered in wind tunnel testing.

CFD The term “computational fluid dynamics (CFD)” refers to computer simulations used to model fluid flows,

such as air or water. These are governed by partial differential equations (continuity, energy, and momentum) that cannot be solved analytically and, therefore, require discretization and must be solved numerically. Discretization is achieved via the use of a mesh, which divides the fluid domain into cells. Specialized solvers then use this discretization, along with the relevant boundary conditions, to calculate flow quantities across the domain. An example of a CFD simulation of a recent F1 car can be found in Ashton et al. [19].

There are different levels of CFD solver fidelity, with the most accurate methods generally demanding the highest computational cost. In general, Euler equation methods and panel methods are considered to have the lowest fidelity and computational cost; these are still used in aerospace-related literature for aerodynamic database generation, but their use in high-level motorsport is no longer commonplace. Reynolds-averaged Navier–Stokes (RANS) models are widely used in both aerospace and motorsport, as they provide sufficiently high-fidelity information without the prohibitively high computational cost associated with higher-fidelity solvers. RANS-based methods can struggle with flows exhibiting large-scale flow separation [20]. Large eddy simulation (LES) equations, on the other hand, are well-suited to flows with large-scale flow separation but require significantly more computational resources due to the inherently transient nature of the simulation and the need to resolve small temporal and spatial scales. A compromise, increasingly used within motorsport and aircraft design, are hybrid RANS-LES methods (the most widely used being the detached eddy simulation (DES) method), which applies LES modeling to resolve large-scale turbulent structures away from the wall and switches to RANS modeling in near-wall regions where modeling the flow is more acceptable [21]. However, even with a DES approach, the computational cost is typically an order of magnitude greater than RANS. Because of this cost, most high-fidelity datasets that will be reviewed in this paper have been generated using a RANS model. In a sport such as F1, where computational resources are constrained, the improvement in fidelity obtained through using a DES model is not necessarily worth the increased time cost, given the reduced number of simulations this would entail, and so a balance must be found between cost and accuracy. As computational resources and hardware (e.g., GPUs) continue to increase, there will be greater use of high-fidelity methods for CFD and aeromap data generation.

On-Track Testing The most realistic environment for analyzing a race car is on a race track. Given that the car has already been built, data from track tests are usually used to validate, update, or augment the data that have been generated using wind tunnels and computer simulations. One of the drawbacks of this type of testing is the reduced control over the relevant independent variables that influence performance: wind speed and direction, tire state, vehicle attitude, and environmental conditions. Unlike wind tunnel testing, these variables are not

slow-moving enough to be considered steady-state for measurement purposes. Furthermore, a full-scale car may have more limited data acquisition capabilities compared with a wind tunnel model.

Surrogate Modeling

Once a dataset has been generated, surrogate models are often employed to learn the input–output mapping in the data, making it possible to obtain output predictions away from the sampled data points. Surrogate modeling methods used in aerodynamic databases include, but are not limited to, polynomial models [22], kriging [23], artificial neural networks (ANNs) [24], and support vector machines (SVMs) [25]. The next section in this paper introduces kriging and neural network–based modeling, as these two approaches are fundamental to most of the data fusion methods that will be reviewed.

Surrogate Modeling Methods

This section introduces the surrogate modeling methods used in combination with data fusion in the reviewed literature. The methods about to be discussed can also be classified as supervised machine learning methods; that is, they are models trained on a dataset comprising features (input parameter values) and labels (output values).

Kriging

This overview outlines the kriging method derivation from Sacks et al. [26] as it most aligns with the data fusion methods presented in the next section. This method derives kriging from a frequentist standpoint, i.e., the derivation is achieved through a constrained minimization of the mean squared error of the kriging predictor. For a Bayesian derivation, the reader is referred to the work of O’Hagan [27]. Both standpoints result in the same kriging predictor being derived. The model input $\mathbf{x}$ is treated as a vector, where each element describes one dimension or feature of the input, and the model output y is treated as a scalar. The kriging approach assumes that the system response $y(\mathbf{x})$ can be treated as the realization of a random function $Y(\mathbf{x})$ modeled by a Gaussian process $Z(\mathbf{x})$ plus some trend μ , which can be either a constant mean value or a regression model:

$$Y(\mathbf{x}) = \mu + Z(\mathbf{x}) \quad (1)$$

The three most common forms of this type of kriging are simple kriging, where μ is assumed to be constant and known; ordinary kriging, where μ is assumed to be constant and unknown; and universal kriging, where a set of regression functions and parameters model the trend. The expression for universal kriging is shown below:

$$Y(\mathbf{x}) = \mathbf{h}(\mathbf{x})^\top \boldsymbol{\beta} + Z(\mathbf{x}) \quad (2)$$

where $\mathbf{h}(\mathbf{x})$ is a known vector of regression functions and $\boldsymbol{\beta}$ is a vector of parameters. In ordinary kriging, $\mathbf{h}(\mathbf{x}) = [1]$.

The remaining part of this section will focus on deriving the best linear unbiased predictor (BLUP) for universal kriging. A BLUP means that the predictor being sought is:

- Best: The predictor mean squared error (MSE) is to be minimized within the class of linear predictors.
- Linear: The predictor is a linear function of the observations, $\mathbf{y}$.
- Unbiased: The expected value of the predictions is to be the same as the expected value of the random process at the prediction point.

Starting with the observed data, also known as training data, where the observed inputs are $\mathbf{X} = [\mathbf{x}_1, \dots, \mathbf{x}_n]^T$ and outputs $\mathbf{y} = [y_1, \dots, y_n]^T$. A linear predictor is a predictor of the form:

$$\hat{y}(\mathbf{x}) = \mathbf{c}(\mathbf{x})^T \mathbf{y} \quad (3)$$

where $\mathbf{c}$ is a vector of weights that depends on the test point, $\mathbf{x}$. Then $\mathbf{y}$ is replaced with its associated random process $\mathbf{Y} = [Y_1, \dots, Y_n]^T$. Next, the unbiasedness condition is enforced by equating the expected value of the predictor to that of the random process:

$$E[\hat{Y}(\mathbf{x})] = E[Y(\mathbf{x})] \quad (4)$$

This leads to the unbiasedness constraint:

$$\mathbf{c}(\mathbf{x})^T \mathbf{H} = \mathbf{h}(\mathbf{x})^T \quad (5)$$

where $\mathbf{h}(\mathbf{x})$ is a column vector of known basis functions evaluated at the test point and $\mathbf{H}$ is the design matrix, which is constructed by stacking the $\mathbf{h}$ vectors for each training point. The random process $Z(\mathbf{x})$ is assumed to have a zero mean and a covariance of the form:

$$\text{cov}(\mathbf{x}, \mathbf{x}') = \sigma^2 K(\mathbf{x}, \mathbf{x}') \quad (6)$$

where σ^2 is the system variance, K is the covariance function or *kernel*, and $\mathbf{x}, \mathbf{x}'$ are two points in the input domain. For example, the power exponential kernel is

$$K(\mathbf{x}, \mathbf{x}') = \prod_j^d \exp\left(-\theta_j |\mathbf{x}_j - \mathbf{x}'_j|^{p_j}\right) \quad (7)$$

where θ_j and p_j are parameters that vary across the d input dimensions.

Next, $\mathbf{y}$ and $\hat{y}(\mathbf{x})$ are treated as random quantities. Following a frequentist approach, the MSE of the predictor averaged over the random process is computed. The vector $\mathbf{c}(\mathbf{x})$ is chosen to minimize:

$$\text{MSE}[\hat{y}(\mathbf{x})] = E\left[\left(\mathbf{c}(\mathbf{x})^T \mathbf{Y} - Y(\mathbf{x})\right)^2\right] \quad (8)$$

Minimizing this leads to the BLUP. This leads to the MSE:

$$\min(\text{MSE}) = \sigma^2 \left[1 + \mathbf{c}(\mathbf{x})^T \mathbf{R} \mathbf{c}(\mathbf{x}) - 2\mathbf{c}(\mathbf{x})^T \mathbf{r}(\mathbf{x}) \right] \quad (9)$$

where $\mathbf{R}$ is the covariance matrix between the training points and $\mathbf{r}$ is the cross-covariance matrix between the training points and the test point $\mathbf{x}$. The constrained minimization of the MSE is accomplished through the use of the Lagrange multipliers $\boldsymbol{\lambda}$. Thus, $\mathbf{c}(\mathbf{x})$ must satisfy:

$$\begin{bmatrix} 0 & \mathbf{H}^T \\ \mathbf{H} & \mathbf{R} \end{bmatrix} \begin{bmatrix} \boldsymbol{\lambda}(\mathbf{x}) \\ \mathbf{c}(\mathbf{x}) \end{bmatrix} = \begin{bmatrix} \mathbf{h}(\mathbf{x}) \\ \mathbf{r}(\mathbf{x}) \end{bmatrix} \quad (10)$$

The BLUP can be determined by inverting the partitioned matrix:

$$\hat{y}(\mathbf{x}) = \mathbf{h}(\mathbf{x})^T \hat{\boldsymbol{\beta}} + \mathbf{r}(\mathbf{x})^T \mathbf{R}^{-1} (\mathbf{y} - \mathbf{H} \hat{\boldsymbol{\beta}}) \quad (11)$$

where $\hat{\boldsymbol{\beta}} = (\mathbf{H}^T \mathbf{R}^{-1} \mathbf{H})^{-1} \mathbf{H}^T \mathbf{R}^{-1} \mathbf{y}$ is the generalized least-squares estimate of the trend parameter vector. Substituting Equation 11 into Equation 9 leads to the MSE estimate for the prediction:

$$\text{MSE}[\hat{y}(\mathbf{x})] = \sigma^2 \left[\left(\mathbf{h}(\mathbf{x})^T \mathbf{r}(\mathbf{x})^T \right) \begin{bmatrix} 0 & \mathbf{H}^T \\ \mathbf{H} & \mathbf{R} \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{h}(\mathbf{x}) \\ \mathbf{r}(\mathbf{x}) \end{bmatrix} \right] \quad (12)$$

The parameters within the correlation function can be optimized, typically, through a maximum likelihood estimate (MLE) or a cross-validation approach. For the MLE approach, the aim is to select the parameters that maximize the likelihood of the observations $\mathbf{y}$. The optimization is achieved by maximizing the concentrated ln-likelihood function, which is derived by taking the natural logarithm of the Gaussian likelihood and removing constant terms:

$$\ln(L) \approx -\frac{n}{2} \ln\left((\mathbf{y} - \mathbf{H} \hat{\boldsymbol{\beta}})^T \mathbf{R}^{-1} (\mathbf{y} - \mathbf{H} \hat{\boldsymbol{\beta}})\right) - \frac{1}{2} \ln|\mathbf{R}| \quad (13)$$

Typically, a numerical optimization method is used to maximize this function; Ollar et al. [28] provided a comparative study of different gradient-based hyperparameter optimization methods for kriging. Alternatively, leave-one-out cross-validation (LOOCV) error can be minimized to optimize the parameters. In this case, the observed data is split into a training set and a validation set. A kriging model is fitted using the reduced training set, and the error between the prediction and the validation point is computed. This process is repeated for all the points, and the mean error is calculated. An example of this approach can be found in the work of Muyskens et al. [29].

Artificial Neural Networks

The fundamental building block of neural networks is the neuron, also known as the unit or node. A neuron receives any number of inputs called *features* and processes them in a nonlinear fashion to return an output. The function that characterizes a neuron is

$$a = \sigma(\mathbf{w}^\top \mathbf{x} + b) \quad (14)$$

where $\mathbf{w}$ is the vector of weights, $\mathbf{x}$ is the input vector of features, and b is the bias term. The activation function, σ , applies a nonlinearity to the output, which enables a neural network to model nonlinear functions [30].

There are many activation functions; however, some of the most common are:

- Sigmoid function: $\sigma(x) = \frac{1}{1 + e^{-x}}$. Often used in binary classification as it maps the input to an output between 0 and 1 [31].
- Hyperbolic tangent (tanh): $\sigma(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$. Maps the output between -1 and 1 [31].
- Rectified Linear Unit (ReLU): $\sigma(x) = \max(0, x)$. Returns the input if the input is positive. Otherwise, it returns 0. ReLU is simple and, therefore, fast to compute and mitigates the vanishing gradient problem [32].

When multiple neurons are connected in layers, they form an ANN. ANNs are able to learn complex relationships between input and output data for regression or classification tasks. Different configurations of neurons are used for different categories of tasks; CNNs are commonly used with grid-like data such as images, and recurrent neural networks (RNNs) are often used with sequential data, where order matters. One of the more straightforward network types is the aforementioned MLP, also known as a fully connected network (FCN) or feedforward neural network (FNN). An example of an MLP's architecture is shown in Figure 4.

The ability of a neural network to predict is dependent on how well its parameters represent patterns in the training data. The action of optimizing the network

parameters is known as training the network; this is done by minimizing a loss function. The loss function evaluates how well a network is learning by comparing the model prediction with respect to a feature vector, with the label; this information can then be used to adjust the model parameters so that the loss function is minimized. There are many loss functions available to use for training a neural network, and typically, the choice of loss function will depend on whether the model is being used for regression or classification. Common loss functions for regression tasks include mean absolute error (MAE) and MSE. The MSE is defined as:

$$L_{\text{MSE}} = \frac{1}{n} \sum_{i=1}^n \|\hat{y}_i - y_i\|^2 \quad (15)$$

where y is the training label, $\hat{y}$ is the predicted label for the i th training example of n [30]. Minimizing the loss function is typically achieved by performing gradient descent, where the gradient of the loss function with respect to the model parameters is used to iteratively update the weights of the model in conjunction with a scaling factor known as the learning rate:

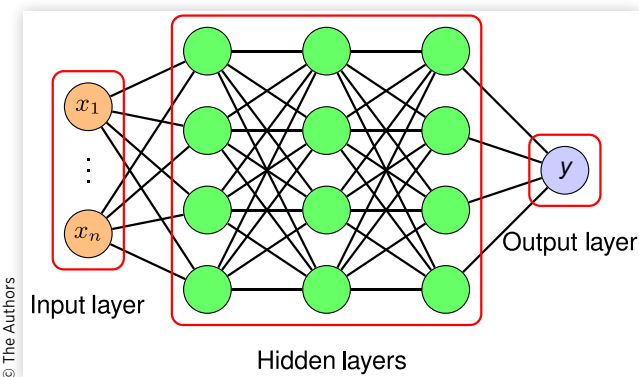
$$\mathbf{w}_{\text{new}} = \mathbf{w}_{\text{current}} - \alpha \nabla_{\mathbf{w}} L(\mathbf{w}_{\text{current}}) \quad (16)$$

where $\mathbf{w}$ is the vector of weights, α is the learning rate, and $\nabla_{\mathbf{w}} L(\mathbf{w}_{\text{current}})$ is the gradient of the loss function evaluated at the current weights [31]. For multilayer networks, this gradient is calculated efficiently using the backpropagation algorithm. In general, selecting a learning rate that is too small will result in a longer training time as more iterations are required to reach the local minimum. Conversely, selecting a learning rate that is too high may result in overshooting the minimum.

In practice, data is processed in batches. A forward and backward pass of all training examples is an epoch. The batch size is the number of examples used in a single forward/backward pass. An iteration corresponds to one such pass (i.e., one weight update). Training can be performed using different strategies: batch gradient descent uses the entire training set for each iteration. This is computationally expensive and memory-intensive for large datasets. Alternatively, stochastic gradient descent (SGD) updates parameters based on the loss calculated over a mini-batch (typically one to a few hundred examples) [30]. This approach is more efficient, and its noisy updates can help the model generalize better. To improve convergence, adaptive learning rate algorithms such as AdaGrad [33], RMSProp [34], and Adam [35] are used.

The performance of a neural network depends heavily on its hyperparameters; configurable settings like the number of layers, the number of neurons per layer, the learning rate, and the number of epochs. To tune these, data is typically split into training, validation, and test sets (e.g., 60%, 20%, 20%). The training set is used to learn the model weights. The validation set is used to evaluate performance and tune the hyperparameters.

FIGURE 4 Architecture of an MLP with three hidden layers.



For small datasets, cross-validation methods like LOOCV can be used instead. Finally, the test set provides a final, unbiased assessment of the finalized model's performance. The model must not be altered after evaluation on the test set to ensure the integrity of its performance metric.

Data Fusion for Aerodynamic Databases

This section reviews data fusion methods used to generate aerodynamic databases for flight simulation. Here, *data fusion* refers to modeling approaches that combine multiple datasets to reduce the overall cost of

data generation without sacrificing model accuracy. This definition encompasses multi-fidelity techniques, which integrate datasets of differing fidelity, as well as transfer learning: a machine learning method in which a model trained on one task is adapted for a similar but distinct task. The section is organized into three parts: surrogate-agnostic, kriging-based, and neural network-based methods. An overview of the methods and their respective references can be found in [Table 1](#).

To define scope, aerodynamic databases used for flight simulation are differentiated from response surfaces for aerodynamic shape optimization (ASO). Aerodynamic databases take positional inputs (e.g., angle of attack and sideslip) and map them to quantities of interest (QoI) such as aerodynamic force and moment coefficients. By contrast, ASO response surfaces take design parameters (e.g., aerofoil control points, camber, twist) as inputs and

TABLE 1 Research efforts on data fusion for aerodynamic databases.

Reference	Year	Surrogate model	Data fusion type
Rajkumar et al. [24]	2002	ANN	ANN fine-tuning
Tang et al. [36]	2003	Linear interpolation	Additive scaling
Tang et al. [7]	2005	Kriging	Multiplicative scaling
Ghoreyshi et al. [8]	2008	Kriging	Multiplicative scaling
Ghoreyshi et al. [37]	2009	Kriging	Input vector augmentation
Han et al. [38]	2010	Kriging	NCM
Kuya et al. [6]	2010	Kriging	AR1 (KOH)
Yamazaki et al. [39]	2011	Kriging	NCM
Da Ronch et al. [40]	2011	Kriging	Input vector augmentation
Han et al. [41]	2012	Kriging	Hierarchical kriging
Cristofaro et al. [42]	2014	Kriging	Input vector augmentation
Tancred et al. [43]	2015	Kriging	NCM
Rumpfkeil et al. [44]	2016	Kriging	Hierarchical kriging (gradient-enhanced)
He et al. [45]	2017	Kriging and RBF	Additive scaling
Lee et al. [23]	2017	Kriging	Multiplicative scaling
Mohammadi-Amin et al. [46]	2018	Kriging	AR1 (KOH)
Tyan et al. [9]	2018	Kriging	Additive scaling
Kaya et al. [47]	2019	Kriging	AR1 (KOH)
Kim et al. [48]	2019	Kriging	Scaling (additive, multiplicative), space mapping
Mukhopadhyaya et al. [49]	2019	Kriging	AR1 (recursive cokriging)
He et al. [10]	2020	ANN	Multi-fidelity neural network
Zhang et al. [50]	2020	Kriging and mixture of experts	AR1 (KOH)
Quinlan et al. [5]	2021	Kriging	AR1 (recursive cokriging)
Arenzana et al. [51]	2021	Kriging	AR1 (recursive cokriging)
Lee et al. [52]	2021	Kriging	AR1 (KOH)
Burchett et al. [53]	2022	Fourier model	Fourier series interpolation
Peng et al. [54]	2023	Kriging	Improved hierarchical cokriging
Pham et al. [55]	2023	Kriging	Extended hierarchical cokriging
Quinlan et al. [56]	2023	Fourier basis, kriging	Fourier basis, AR1 (recursive cokriging)
Burchett et al. [57]	2023	Kriging	AR1 (KOH)
Scoggins et al. [58]	2023	Kriging	Multihierarchy Gaussian processes
Yang et al. [59]	2024	Kriging	Pointwise variance optimization
Rataczak et al. [4]	2024	Kriging (multi-output)	AR1 (recursive cokriging)
Ni et al. [11]	2024	ANN	Fine-tuning
Xu et al. [60]	2024	Kriging	Hierarchical kriging

map them to similar QoI. Because the target application is motorsport aeromaps, and the motorsport-specific literature is sparse, this review is restricted to flight simulation aerodynamic databases, which most closely mirror the aeromap formulation; ASO works are cited only when their methods directly inform aerodynamic database construction. For machine learning methods in ASO, see Li et al. [61]. Accordingly, this review includes studies whose inputs are positional variables and excludes those whose inputs are design variables.

Surrogate-Agnostic Methods

The first category of data fusion methods will be referred to as *surrogate-agnostic*, this category comprises scaling functions and input vector augmentation. These methods do not rely on a specific modeling technique to achieve data fusion.

Scaling Functions Variable complexity modeling (VCM) was introduced by Hutchison et al. [62] for design optimization, which was based on the global–local approximation method [63]. The method uses a scaling or *bridge* function $\sigma(\cdot)$ that describes the relationship between the high- and low-fidelity data $f_H(\cdot)$, $f_L(\cdot)$ by taking the ratio of the data at a specific sampling point x_0 :

$$\sigma(x_0) = \frac{f_H(x_0)}{f_L(x_0)} \quad (17)$$

Then, a quadratic response surface model is fitted to the ratio values at each sample point using a least-squares algorithm:

$$\sigma(x) = c_0 + \sum_{i=1}^p c_i x_i + \sum_{i,j=1}^p c_{ij} x_i x_j \quad (18)$$

where x_i represents the input variables, c_i are the polynomial coefficients, and p is the number of input parameters. The enhanced function is then calculated by multiplying the bridge function $\sigma(x)$ by the low-fidelity data, leading to a *multiplicative* scaling function:

$$f(x) \approx \sigma(x) f_L(x) \quad (19)$$

Tang et al. [7] highlighted two issues with the multiplicative scaling function. First, if $f_L(x)$ is very small or approaches zero, the scaling function $\sigma(x)$ can become unrealistically large or fail altogether. Second, the choice to use a quadratic function to model the scaling factor would prove inaccurate if the data did not correspond to a linear or quadratic trend. To address these issues, the authors proposed defining the scaling function as an increment rather than a ratio. Lewis and Nash [64] also proposed the same *additive* scaling function in the context of optimization. To address the limitations of a quadratic model, the increment function, β , was modeled using

kriging, improving the flexibility and accuracy of the scaling function. The increment function values are calculated as follows:

$$\beta(x_0) = f_H(x_0) - f_L(x_0) \quad (20)$$

This definition yields the additive scaling function:

$$f(x) \approx \beta(x) + f_L(x) \quad (21)$$

where $\beta(x)$ is a kriging model trained with the increment data. This form of additive scaling function was used to generate aerodynamic databases with one, two, and three input variables (angle of attack, Mach number, and sideslip), with low- and high-fidelity data obtained from Euler and Navier–Stokes simulations, respectively. In these three test cases, data fusion (in combination with an adaptive sampling method for the second and third cases) reduced the computational cost to 31%, 32%, and 25%, respectively, of the equivalent high-fidelity simulation generated with a full-factorial sampling plan, while maintaining a high level of accuracy.

Though simple, increment functions have been used to good effect in aerodynamic database modeling, with subsequent implementations by Ghoreyshi et al. [8], where they were applied to a UAV configuration. In this application, various low-fidelity datasets, digital DATCOM and VLM code, were fused with a small number of RANS solutions. The authors found that using both low-fidelity datasets along with only one RANS sample resulted in a model that showed strong agreement with a test set of RANS (HF) observations. Lee et al. [23] applied this approach to a gliding vehicle aerodynamic database using VLM and RANS solutions as the low- and high-fidelity datasets, respectively. Part of the success of this method is dependent on the relationship between the high-fidelity and low-fidelity datasets, along with where the high-fidelity data is sampled. If knowledge of the full HF dataset is available, then it is straightforward to optimize sample placement to fully capture the relationship between LF and HF datasets. However, this knowledge is not typically available, meaning that a small number of HF samples could easily corrupt the multi-fidelity model. Another limitation of this approach is that it does not provide an uncertainty estimate when used in conjunction with kriging, making scaling functions less ideal for uncertainty-based adaptive sampling methods.

Input Vector Augmentation A form of input vector augmentation was implemented by Ghoreyshi et al. [37] to achieve data fusion for aerodynamic tables. The method works by initially training a regression model on the available LF data and evaluating it at the HF sample locations based on the notion that the high-fidelity data y_H are a function of both the inputs X and the low-fidelity data y_L :

$$y_H = f(X, y_L) \quad (22)$$

Given the training features $\mathbf{x}_i$ and labels y_i , the low-fidelity predictor is denoted as $\hat{y}_L$. The HF feature vector is augmented by the predicted response using the model trained on the LF data:

$$\mathbf{x}_i^{\text{aug}} = [x_1, \dots, x_n, \hat{y}_L(\mathbf{x}_i)] \quad (23)$$

where $\mathbf{x}_i$ is the i th original feature vector and $\hat{y}_L(\mathbf{x}_i)$ is the surrogate model trained on the LF dataset and evaluated at the HF sample locations. x_1 to x_n are the relevant input parameters, also known as features. The augmented feature vector can then be used to train a multi-fidelity surrogate, where evaluation is performed on an augmented test vector of the same form as the training vector. This augmented approach allows the surrogate to learn information from both the LF and HF datasets. The authors applied this method to update an aerodynamic dataset based on the availability of an initial one. Here, the geometry was incremented from an initial design. A new dataset of 29 geometries was generated using different wing sweep, wing and tail aspect ratios, fuselage lengths and diameters. This data fusion reduced the number of high-fidelity samples needed to just 10. For comparison, generating the aerodynamic database from scratch required 35 high-fidelity samples.

Kriging-Based Methods

The treatment of multiple cross-correlated variables for regression was initially developed within mining geostatistics [65, 66]. The term given to this methodology is cokriging. Cokriging gained popularity within surrogate modeling following the work of Kennedy and O'Hagan [67], where the approach was extended to modeling for computer experiments. Since then, various adaptations of multi-fidelity kriging models have been proposed. This section discusses the methodology and the unique characteristics of each approach, along with the applications within aerodynamic database generation. For a comprehensive review of cokriging methods for engineering surrogate modeling, the reader is referred to the work of Brevault et al. [68].

Autoregressive Model In 2000, Kennedy and O'Hagan [67] proposed a first-order autoregressive (AR1) method that approximates a higher-fidelity output using a scaled lower-fidelity model in addition to a correction term:

$$y_t(\mathbf{x}) = \rho_{t-1} y_{t-1}(\mathbf{x}) + \delta_t(\mathbf{x}) \quad (24)$$

where $y_t(\cdot)$ is the high-fidelity model prior, ρ_{t-1} is the scaling factor that quantifies the correlation between the high- and scaled low-fidelity data, $y_{t-1}(\cdot)$ is the lower-fidelity data

modeled by a Gaussian process, and $\delta_t(\cdot)$ is a separate Gaussian process used to model the discrepancy between the high- and low-fidelity data. To distinguish this approach from the subsequent AR1 methods, this model will be referred to as the *KOH* model. For two levels of fidelity, the model is denoted as

$$y_2(\mathbf{x}) = \rho_1 y_1(\mathbf{x}) + \delta_2(\mathbf{x}) \quad (25)$$

For simplicity, the remaining explanation for this method will assume only two levels of fidelity. Like single-fidelity kriging, cokriging requires the construction of a covariance matrix. This covariance matrix, $\mathbf{R}$, is an extension of the single-fidelity covariance matrix but with additional correlation terms for the LF data and the discrepancy data. To obtain the covariance matrices for the various datasets, the method applies a form of Markov property:

$$\text{cov}[y_2(\mathbf{x}), y_2(\mathbf{x}') | y_1(\mathbf{x})] = 0, \quad \forall \mathbf{x}' \neq \mathbf{x} \quad (26)$$

This enforces the assumption that no more information can be obtained at $y_2(\mathbf{x})$ from the lower-fidelity data $y_1(\mathbf{x})$ for $\forall \mathbf{x}' \neq \mathbf{x}$. This leads to the covariance matrix, where the top-left block corresponds to the low-fidelity covariance, while the bottom-right block corresponds to the high-fidelity covariance. The off-diagonal blocks correspond to the correlation between the high- and low-fidelity data:

$$\mathbf{R} = \begin{bmatrix} \sigma_1^2 K_1(X_1, X_1) & \rho_1 \sigma_1^2 K_1(X_1, X_2) \\ \rho_1 \sigma_1^2 K_1(X_2, X_1) & \rho_1^2 \sigma_1^2 K_1(X_2, X_2) + \sigma_2^2 K_2(X_2, X_2) \end{bmatrix} \quad (27)$$

where K_t is the kernel, X_t are the input data and σ_t^2 is the system variance for a specific fidelity level t . As a result of the additional correlation terms, there are twice as many model hyperparameters to determine, in addition to the scaling factor, ρ . Hyperparameter optimization for the low-fidelity dataset can be achieved by maximizing the concentrated ln-likelihood of the dataset. The Markov property defined in Equation 26 implies that the high-fidelity hyperparameters only depend on the high-fidelity data. Consequently, these hyperparameters are found by minimizing the concentrated ln-likelihood of the *discrepancy* dataset, given by

$$d_2 = y_2 - \rho_1 y_1(X_2) \quad (28)$$

The mean function of the posterior acts as the cokriging predictor at a test point:

$$\hat{y}(\mathbf{x}) = \mathbf{h}'(\mathbf{x})^\top \hat{\boldsymbol{\beta}} + \mathbf{r}(\mathbf{x})^\top \mathbf{R}^{-1}(\mathbf{y} - \mathbf{H}\hat{\boldsymbol{\beta}}) \quad (29)$$

The matrices that make up the predictor are as follows:

$$\begin{aligned}
 \mathbf{h}'(\mathbf{x})^\top &= \begin{bmatrix} \rho \mathbf{h}(\mathbf{x})^\top & \mathbf{h}(\mathbf{x})^\top \end{bmatrix} \\
 \mathbf{H} &= \begin{bmatrix} \mathbf{h}(\mathbf{x}_1^{(1)})^\top & 0 \\ \vdots & \vdots \\ \mathbf{h}(\mathbf{x}_{n_1}^{(1)})^\top & 0 \\ \rho \mathbf{h}(\mathbf{x}_1^{(2)})^\top & \mathbf{h}(\mathbf{x}_1^{(2)})^\top \\ \vdots & \vdots \\ \rho \mathbf{h}(\mathbf{x}_{n_2}^{(2)})^\top & \mathbf{h}(\mathbf{x}_{n_2}^{(2)})^\top \end{bmatrix} \\
 \hat{\boldsymbol{\beta}} &= \begin{bmatrix} \hat{\beta}_1 \\ \hat{\beta}_2 \end{bmatrix} = (\mathbf{H}^\top \mathbf{R}^{-1} \mathbf{H})^{-1} \mathbf{H}^\top \mathbf{R}^{-1} \mathbf{y} \\
 \mathbf{r}(\mathbf{x})^\top &= \begin{bmatrix} \rho_1 \sigma_1^2 K_1(\mathbf{x}, \mathbf{x}_1) & \rho_1^2 \sigma_1 K_1(\mathbf{x}, \mathbf{x}_2) \\ + \sigma_2^2 K_2(\mathbf{x}, \mathbf{x}_2) \end{bmatrix} \\
 \mathbf{y} &= \begin{bmatrix} \mathbf{y}_1 \\ \mathbf{y}_2 \end{bmatrix}
 \end{aligned} \tag{30}$$

where $\mathbf{h}'(\mathbf{x})^\top$ is a vector of basis functions, $\hat{\boldsymbol{\beta}}$ is a vector of regression coefficients, determined through generalized least squares, $\mathbf{H}$ is a design matrix of basis functions, $\mathbf{r}(\mathbf{x})^\top$ is the cross-covariance between the training data and the test point, and $\mathbf{y}$ is the vector of stacked training data.

In scenarios with many samples or multiple fidelity datasets, the KOH method can become computationally demanding because the size of the covariance matrix $\mathbf{R}$ grows with both the number of samples and the number of datasets. To address this limitation, Le Gratiet and Garnier proposed recursive cokriging [69] as a more efficient way of implementing the AR1 approach. The increased efficiency comes from recursively treating the autoregressive structure, thereby making it possible to train each level of fidelity individually, thus reducing the cost associated with inverting large matrices. The predictor for recursive cokriging is

$$\begin{aligned}
 \hat{y}_t(\mathbf{x}) &= \rho_{t-1} \hat{y}_{t-1}(\mathbf{x}) + \mathbf{f}_t(\mathbf{x})^\top \boldsymbol{\beta}_t \\
 &+ \mathbf{r}_t(\mathbf{x}) \mathbf{R}_t^{-1} (\mathbf{y}_t - \rho_{t-1} \hat{y}_{t-1}(\mathbf{x}_t) - \mathbf{F}_t \boldsymbol{\beta}_t)
 \end{aligned} \tag{31}$$

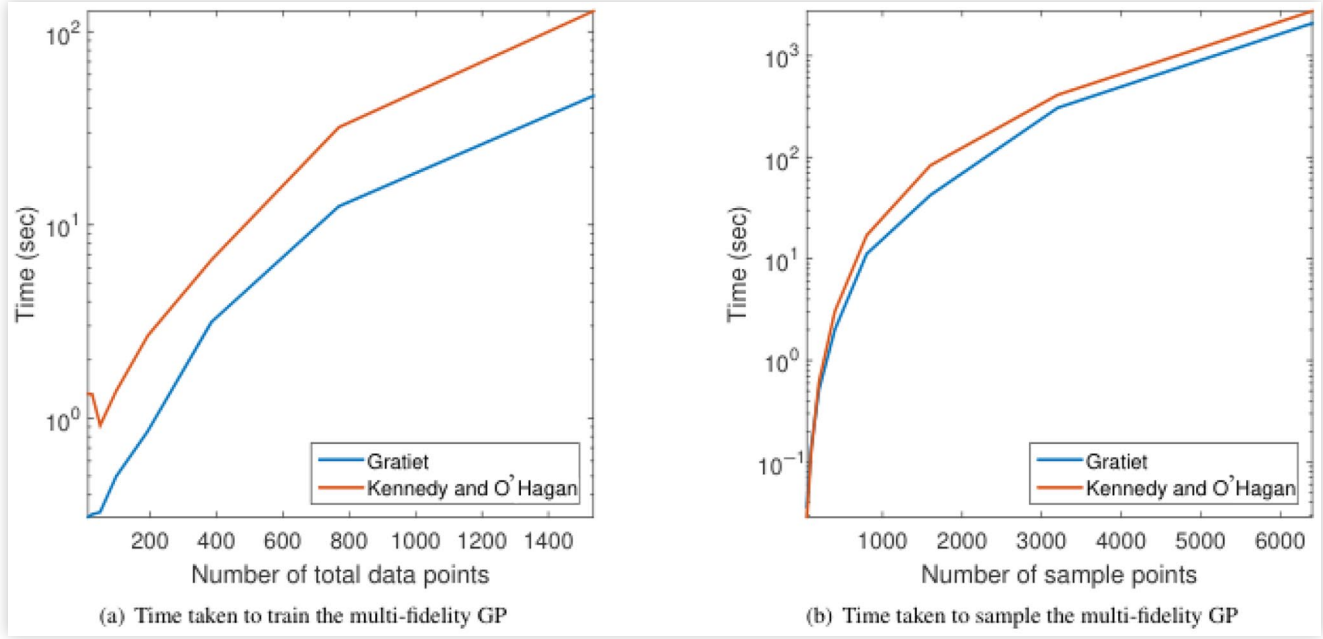
where $\mathbf{f}$ and $\mathbf{F}$ replace the vector of basis functions and design matrix with the recursive equivalent that models the mean trend of the residual. Initially, the process trains a model on the lowest fidelity dataset. Then, the higher-fidelity models are trained on the difference between the observations at the current fidelity level and the observations at the same training points for the previous fidelity

level. In addition, the covariance matrix $\mathbf{R}_t = [\sigma_t^2 K_t(\mathbf{x}_t, \mathbf{x}_t)]$, implementing only one level of data fidelity at a time. This approach has the benefit of only needing to invert s smaller covariance matrices rather than one large one. Through this method, the computational expense is reduced from $\mathcal{O}\left(\left[\sum_{i=1}^s m_i\right]^3\right)$ to $\mathcal{O}\left(\left[\sum_{i=1}^s m_i^3\right]\right)$ where s is the number of fidelity levels.

Mukhopadhyaya et al. [49] used both the KOH and recursive AR1 approaches to generate an aerodynamic database for the NASA common research model. In this work, the authors implemented three levels of data fidelity: VLM simulations, RANS simulations, and wind tunnel data. The authors compared the KOH implementation of the AR1 approach with recursive cokriging and noted that computational time savings increased with the number of datasets and dimensionality. The results from this time complexity analysis are shown in Figure 5. Mohammadi-Amin et al. [46] applied the KOH approach to generate an aerodynamic database for the Orion reentry vehicle. The authors found that the computational time required for all the necessary HF and LF simulations could be reduced from 75 h to 24 h (68%) compared with a single-fidelity model. In addition, if the task was updating a previous database, the reduction increased to 12 h (84%). Kaya et al. [47] used the KOH method to generate a one-dimensional aerodynamic model for a UCAV. LF data were generated using an Euler solver, and HF data were generated using a RANS solver. To avoid the need for extrapolation, initial HF samples were located at the boundaries and mid-points of the domain, and new HF points were added adaptively based on a weighted average of the maximum error, where the error was calculated as the absolute value of the difference between the multi-fidelity model and the high-fidelity model. The authors compared an ordinary and universal form of the KOH method, the former assuming a constant mean and the latter assuming a global polynomial trend. The universal KOH was found to outperform ordinary KOH, requiring one fewer HF sample, out of a total of seven, to achieve similar accuracy.

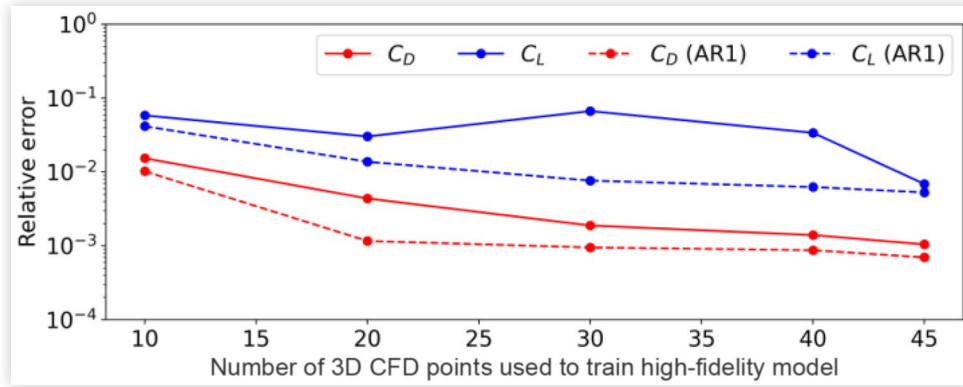
Rataczak et al. [4] used a multi-output implementation of recursive cokriging to leverage the correlation between output parameters for a hypersonic vehicle. The authors used an intrinsic coregionalization model [70] to correlate the model outputs. The authors then compared the multi-fidelity model to a single-fidelity model for generating an aerodynamic database. The multi-fidelity model demonstrated a marked performance improvement, requiring only half as many high-fidelity training points as the single-fidelity model. These results are shown in Figure 6. This work demonstrated that multi-output kriging can be effective when applied to aerodynamic database generation. However, the authors did not compare the performance of separate single-output models with the multi-output model to verify whether the multi-output method provided improved accuracy over the single-output methods.

FIGURE 5 Comparison of training and querying time for Kennedy and O'Hagan's method and Le Gratiet's method (Recursive cokriging).



Reprinted with permission from Ref. [49]. © 2019 Begell House, Inc.

FIGURE 6 Error comparison for single-fidelity Gaussian process regression model and AR1 multi-fidelity model for lift and drag coefficient at different numbers of CFD training points.



Reprinted with permission from Ref. [4]. © 2024 the American Institute of Aeronautics and Astronautics

New Cokriging Method In 2010, Han et al. [38] proposed an alternative approach to AR1 cokriging as a direct extension to geostatistical cokriging but targeted toward computer simulations. This method was introduced as the *new cokriging method* and will be referred to as NCM in this work. NCM uses a different covariance matrix formulation from AR1, and the proposed model form follows the principle of ordinary kriging, i.e., the trend component of the predictor is formed of a constant mean. The HF and LF observations are stacked in one column vector, then an augmented correlation matrix is constructed. The NCM approach has the benefit of being more straightforward to implement and having more in common with single-fidelity kriging. The predictor is given as

$$\hat{y}_2(\mathbf{x}) = \mathbf{h}^T \hat{\boldsymbol{\beta}} + \mathbf{r}^T(\mathbf{x}) \mathbf{R}^{-1}(\mathbf{y} - \mathbf{H} \hat{\boldsymbol{\beta}}) \quad (32)$$

where

$$\begin{aligned} \mathbf{h} &= \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \quad \hat{\boldsymbol{\beta}} = \begin{bmatrix} \hat{\beta}_2 \\ \hat{\beta}_1 \end{bmatrix} = (\mathbf{H}^T \mathbf{R}^{-1} \mathbf{H})^{-1} \mathbf{H}^T \mathbf{R}^{-1} \mathbf{y} \\ \mathbf{r} &= \begin{bmatrix} r_2 \\ r_1 \end{bmatrix}, \quad \mathbf{R} = \begin{bmatrix} \mathbf{R}^{22} & \mathbf{R}^{21} \\ \mathbf{R}^{12} & \mathbf{R}^{11} \end{bmatrix} \\ \mathbf{y} &= \begin{bmatrix} y_2 \\ \frac{\sigma_2}{\sigma_1} y_1 \end{bmatrix}, \quad \mathbf{H} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \end{aligned} \quad (33)$$

with $\mathbf{r}_1, \mathbf{r}_2$ being the covariance vectors between the low- and high-fidelity data and the test point $\mathbf{x}$. $\mathbf{R}^{11}, \mathbf{R}^{12}, \mathbf{R}^{21}, \mathbf{R}^{22}$ are the covariance and cross-covariance matrices between the low- and high-fidelity data, and $\mathbf{H}$ is the design matrix for a constant mean. The terms in the $\mathbf{h}$ vector are used to select only the β_2 term from the generalized least-squares predictor; this is the term that relates to the high-fidelity data.

As with the AR1 method, the NCM uses a joint correlation matrix to correlate the datasets, the difference being that this joint correlation matrix does not assume the same autoregressive structure. Instead, a parameter transformation is performed to remove the system variances σ_1 and σ_2 from the covariance matrix and into the stacked output vector as a scaling factor on $\mathbf{y}_1$. This reduces the probability of encountering numerical instabilities when the magnitudes of σ_1 and σ_2 are significantly different. Another benefit stems from the lack of assumption of a Markov property, resulting in no longer requiring nested sampling plans. Furthermore, all hyperparameters can be optimized in one step, in contrast to the multi-step procedure used in the KOH method. The authors also note that function gradient information can be implemented more easily, opening up options for a gradient-enhanced kriging model. The one drawback noted by the authors is the small increase in computational expense of NCM relative to the KOH method.

Yamazaki and Mavriplis [39] used a similar method to NCM to generate aerodynamic databases for the common research model and also applied the technique to a drag minimization problem for a 2D aerofoil. In this work, the authors successfully implemented gradient information using the method detailed in [71] to obtain a faster reduction of the objective function value. The gradient information was incorporated into the model via additional terms in the covariance matrix. Tancred and Rumpfkeil [43] used the same multi-fidelity method as Yamazaki and Mavriplis to generate an aerodynamic database for a generic hypersonic vehicle, fusing LF data from a modified Newtonian method with HF data from an Euler solver. The database had Mach number and α as inputs and C_L and C_D as outputs. The authors obtained the initial high-fidelity samples at the corners and center of the Mach- α domain and the initial low-fidelity samples through LHS. An intermediate multi-fidelity model was then fitted, and adaptive sampling was applied to place new high-fidelity samples. The analysis showed that increasing the amount of low-fidelity data (0, 50, 200 samples) tended to improve the accuracy of the response surface. However, in one case with 10 HF samples, adding 50 LF samples actually reduced the modeling accuracy. This was remedied by increasing the number of LF samples to 200, but this raised the issue of LF samples interfering with the model for some sample sizes. Furthermore, the authors noted that once the number of HF samples was increased over 20, the benefit of introducing LF samples was diminished.

Hierarchical Kriging As discussed earlier, universal kriging refers to a Gaussian process regression model

with a general trend model. This trend model is typically defined using a linear or quadratic polynomial. In 2012, Han and Gortz [41] proposed hierarchical kriging (HK), a method that uses a secondary Gaussian process to model the trend. Because of how the multi-fidelity datasets are combined, AR1 and NCM models output an implausible uncertainty estimate which oscillates about the low-fidelity data points. This behavior can lead to ineffective sample placement when using adaptive sampling based on this uncertainty estimate. The HK predictor is constructed in such a way as to provide a plausible uncertainty estimate. To build the predictor, first, a surrogate is built using the low-fidelity data

$$Y_1(\mathbf{x}) = \hat{\beta}_1 + Z_1(\mathbf{x}) \quad (34)$$

where $\hat{\beta}_1$ is an unknown constant and $Z_1(\mathbf{x})$ is a stationary Gaussian process. The low-fidelity kriging predictor is given as

$$\hat{y}_1(\mathbf{x}) = \hat{\beta}_1 + \mathbf{r}_1(\mathbf{x})^\top \mathbf{R}_1^{-1}(\mathbf{y}_1 - \hat{\beta}_1 \mathbf{1}) \quad (35)$$

where $\mathbf{R}_1$ is the covariance matrix of low-fidelity observations, $\mathbf{r}_1$ is the correlation vector representing the correlations between the test point and the observed points, and $\mathbf{1}$ is a vector of ones that is the same length as the number of observations. Then, the high-fidelity function is modeled as

$$Y_2(\mathbf{x}) = \hat{\beta}_2 \hat{y}_1 + Z_2(\mathbf{x}) \quad (36)$$

The expression for the high-fidelity predictor is given as

$$\hat{y}_2(\mathbf{x}) = \hat{\beta}_2 \hat{y}_1(\mathbf{x}) + \mathbf{r}_2^\top(\mathbf{x}) \mathbf{R}_2^{-1}(\mathbf{y}_2 - \hat{\beta}_2 \mathbf{H}) \quad (37)$$

where

$$\begin{aligned} \mathbf{H}_t &= \left[\hat{y}_{t-1}(\mathbf{x}^{(1)}), \dots, \hat{y}_{t-1}(\mathbf{x}^{(n)}) \right]^\top \\ \mathbf{R}_t &= \left[K_t(X_t, X_t') \right] \\ \mathbf{r}_t &= \left[K_t(X_t, \mathbf{x}) \right] \\ \hat{\beta}_t &= \left(\mathbf{H}_t^\top \mathbf{R}_t^{-1} \mathbf{H}_t \right)^{-1} \mathbf{H}_t^\top \mathbf{R}_t^{-1} \mathbf{y}_t \end{aligned} \quad (38)$$

for fidelity level t . Compared to AR1 methods, HK does not rely on a cross-correlation term between the low- and high-fidelities, resulting in a less complex model with a smaller covariance matrix than the KOH approach. Because of this formulation, the model outputs a more plausible uncertainty estimate than the AR1 method. Importantly, this improved uncertainty estimate is achieved with a similar computational cost to recursive cokriging.

HK has undergone various adaptations since it was first proposed. In 2019, Zhou et al. [72] adapted the

method to enable it to accept both nested and non-nested multi-fidelity sampling plans. This adapted method was termed generalized hierarchical co-kriging (GCK). The authors compared GCK against the KOH method and HK, as well as another method called hybrid kriging scaling (HKS), on various engineering problems, one of which was an aerodynamic database for a 2D aerofoil. The authors found that GCK performed identically to KOH when using nested samples. Furthermore, for non-nested samples, they found that GCK's improvement relative to KOH increased as the number of LF samples decreased. Conversely, HK performed better than GCK when a small number of LF samples were used, and GCK improved over HK when the number of LF samples was large.

Peng et al. [54] proposed an improved HK (IHK) approach and applied the framework for unsteady modeling by [73] to adapt HK for unsteady flow modeling. Pham et al. [55] proposed an extended HK (EHK) method and used it to generate an aerodynamic database for an eVTOL aircraft. Xu et al. [74] introduced an expert's experience-informed HK (EEI-HK) method to address the limited extrapolation capabilities and physical interpretability of standard kriging-based methods. Rather than using a low-fidelity model as the trend function, this approach defines an empirical trend based on expert experience and physical knowledge of the system. The work proposes trigonometric series, polynomial incremental models, and state-space models as some of the experience-informed trends for the kriging model. This approach was validated on a missile geometry and hypersonic flight vehicle geometry, and the authors found that EEI-HK outperformed ordinary and universal kriging in both interpolation and extrapolation, across lift, drag, and moment coefficient outputs.

Multihierarchy Gaussian Process Regression Scoggins et al. [58] proposed *multihierarchy* Gaussian process regression (MHGPR), a generalized form of the AR1 framework which does not require a strict ranking of datasets. Instead, this model structure relies on an idealized nominal baseline dataset along with several correction functions representing different sources of deviation from the nominal dataset. The method assumes that the true function $f(\mathbf{x})$ can be described in the following form:

$$f(\mathbf{x}) = f_0(\mathbf{x}) + \sum_{i=1}^c w_i \Delta f_i(\mathbf{x}) \quad (39)$$

where w_i are weights that control the level of influence of each correction function $\Delta f_i(\mathbf{x})$ and $f_0(\mathbf{x})$ is the nominal function, modeled by a zero-mean Gaussian process $f_0(\mathbf{x}) = \delta_0(\mathbf{x}) \sim \mathcal{GP}(0, K_0)$, with kernel K_0 . The correction functions $\Delta f_i(\mathbf{x})$ are defined by the offset between the nominal function $f_0(\mathbf{x})$ and the unknown underlying function $f_i(\mathbf{x})$:

$$\Delta f_i(\mathbf{x}) = f_i(\mathbf{x}) - f_0(\mathbf{x}), \quad \forall i \in \{1, \dots, c\} \quad (40)$$

where each $f_i(\mathbf{x})$ is assigned a two-fidelity Gaussian process prior:

$$f_i(\mathbf{x}) = \rho_i f_0(\mathbf{x}) + \delta_i(\mathbf{x}) \quad (41)$$

$$\delta_i(\mathbf{x}) \sim \mathcal{GP}(0, K_i) \quad (42)$$

Following the determination of the fidelity relationships, the Gaussian process prior on the unknown function is conditioned on the training data to determine the predicted mean and variance. Determining the weights w_i can be done using a truth dataset, or, in the case of no truth data, the weights can be assigned prior distributions. The MHGPR was applied to an aerodynamic database test case for the Orion spacecraft crew module by training the model on three Orion Crew Module wind tunnel datasets, with a fourth dataset acting as the test set. The first dataset was treated as the nominal condition, and the other two datasets were treated as corrective datasets with the aim of capturing the effect of roughness and asymmetries in the geometry. The authors noted good agreement between the predictions and the test set. Furthermore, the interpretable nature of the model was used to analyze various characteristics of the modeled aerodynamic coefficients. For example, larger correction weights associated with surface roughness for the drag and pitching moment coefficients suggested that these coefficients were more dependent on surface roughness than the lift coefficient, which had smaller correction weights associated with those correction functions.

Distribution-Based Weight Control Yang et al. [59] introduced a pointwise variance optimization strategy called Distribution-Based Weight Control (DBWC) for optimizing the fusion of multi-fidelity datasets. This method leverages heteroscedastic Gaussian processes (HGPs) to minimize local predictive uncertainty via the optimization of weight allocation to each fidelity level. The model outputs relating to each fidelity are fused by taking their weighted sum

$$\hat{y}_H(\mathbf{x}) = \sum_{i=1}^m w_i \hat{y}_i(\mathbf{x}) \quad (43)$$

where $\hat{y}_H$ is the HF prediction, w_i is the weight coefficient, and $\hat{y}_i(\mathbf{x})$ is the prediction for the i th fidelity level model. The predictive variance of the i th fidelity level model is

$$\hat{\sigma}_i^2(\mathbf{x}) = \hat{\sigma}_{\text{HGP},i}^2(\mathbf{x}) + \hat{\sigma}_{f,i}^2(\mathbf{x}) \quad (44)$$

where $\hat{\sigma}_{\text{HGP},i}^2$ is the HGP posterior variance and $\hat{\sigma}_{f,i}^2(\mathbf{x})$ is a user-specified fidelity variance, which is based on expert knowledge or user experience. To determine the weight coefficients, the weights are optimized to solve

$$\min_{w_i(\mathbf{x})} \sum_{j=1}^k \sum_{i=1}^m w_i(\mathbf{x})^2 \sigma_i^2(\mathbf{x}_{ij}) \quad (45)$$

$$\text{s.t. } w_i(\mathbf{x}) > 0, \quad \sum_{i=1}^m w_i(\mathbf{x}) = 1$$

where $\mathbf{x}$ is a prediction point, $\mathbf{x}_{ij}$ is a sample point, k is the number of nearest sample points to $\mathbf{x}$, and m is the number of fidelity levels. The solution to this optimization can be obtained in closed form via the Lagrange multiplier method. The authors then propose a weight adjustment scheme that modifies the weight coefficients depending on how well the region around the prediction point is covered by that fidelity level's samples.

DBWC was applied to a Branin function test case to evaluate how the similarity between an LF and an HF dataset affected model performance. Subsequently, the DBWC method was used to fuse drag coefficient data for a Sanger-like vehicle. 20 HF and 78 LF samples were used to cover the two-dimensional input space of Mach number and angle of attack. Several models were compared with the DBWC method, including the variance-weighted sum method from [75], cokriging, a multi-fidelity neural network, and improved HK. The DBWC method was found to outperform all the other methods in RMSE and maximum absolute error.

ANN-Based Methods

The third and final category of data fusion methods relates to neural networks. The following methods implement data fusion either using the training mechanism of neural networks or their modular nature.

Fine-Tuning Transfer learning through fine-tuning is a type of *inductive* transfer learning [76] and has been used to fuse multi-fidelity datasets in ASO [77, 78]. The motivation for pre-training a network on LF data, freezing a subset of the model layers, and then fine-tuning on HF data is that the network learns to model the high-fidelity data, while still retaining knowledge of the low-level features (learned from the LF data) that are still relevant to the HF dataset. This type of transfer learning is included as a form of data fusion, as it involves combining two multi-fidelity datasets to improve predictive performance.

Neural networks have been used since the 1990s as a means to model aircraft aerodynamic coefficients [79–81]. However, one of the first implementations of neural network-based data fusion is in the work of Rajkumar et al. [24]. In this technical note, the authors detail a method where a three-layer MLP is trained on a rich dataset comprising 233 training pairs from a wind tunnel test. The rich dataset allows the network to learn the general pattern characterizing a particular aerodynamic coefficient, which in this case is the lift coefficient with respect to angle of attack and Mach number. Subsequently, the model is retrained on a sparse dataset of nine training pairs so that this sparse dataset can be more accurately interpolated.

Wu et al. [82] used fine-tuning with LF and HF CFD data for the purpose of ASO. Though this is not strictly

the same as generating a conventional aerodynamic database, the model was used to model lift and drag coefficients with respect to some design variables. For an aerodynamic database, these design variables would be replaced with state variables such as angle of attack and Mach number. Transfer learning is achieved by pre-training the network on an LF dataset generated with a coarse mesh, then two of the three network layers are frozen to preserve the high-level features, and finally, the model is retrained on the HF fine mesh data. Liao used a similar approach, but with convolutional neural networks [78], and Leng extended the method to three levels of fidelity [83]. Outside of aerospace, this fine-tuning method has also been used in chemical modeling [84] and battery modeling [85].

In the work of Ni et al. [11], the authors implemented an aerodynamic modeling method using transfer learning and a dendrite net (DD), a white-box model initially proposed by Liu and Wang [86], due to the reported improved performance when tasked with a limited training dataset and the system identification properties inherent in a white-box model. The DD architecture is based on the function of a biological dendrite, and the formula that represents its architecture is given by:

$$Y = W^{L,L-1} \left[\dots W^{l,l-1} \left(\dots W^{2,1} \left(W^{1,0} X \odot X \right) \dots \right) \right] \quad (46)$$

$L \in \mathbb{N}^+$

where X and Y are the input and outputs, respectively, and $W^{l,l-1}$ is the weight matrix between modules $l-1$ and l . The $\odot$ symbol represents element-wise multiplication between the matrices. Ni et al. were motivated by the work of Huang et al. [87], who had previously used a DD as an aerodynamic surrogate model and demonstrated that DDs could outperform an MLP model in both accuracy and training time within this application. Unlike most neural networks, DDs use a white-box algorithm, which allows for feature extraction. The authors used this property to extract the system features to generate a polynomial aerodynamic model. First, the model is trained on a dense dataset of feature-label pairs of aerodynamic data from a source dataset. Then, the model is retrained on a sparser target dataset. The trained model can then be used to obtain the polynomial coefficients via the least-squares method:

$$A_L = \left(X_L^T X_L \right)^{-1} X_L^T Y_L \quad (47)$$

where A_L denotes the fitting coefficient matrix and X_L , Y_L represent the inputs and outputs. Ni et al. [11] used this approach to build an aerodynamic database for an SN-LS-T configuration missile. The DD model is compared with an MLP model and a long-short-term memory (LSTM) model. The model outputs are: pitching moment coefficient C_{PM} , normal force coefficient C_{NF} and axial force coefficient C_{AF} . The source dataset comprises 120 training samples and 20 test samples, and the models are fine-tuned on 30 training samples (25%) of the target quantity.

This amount of target samples is found to allow the target model to perform similarly to the source model. The DD model exhibits lower test RMSE and MAE values for the three outputs (C_{PM} , C_{NF} , C_{AF}). Subsequently, the DD model is used to extract the polynomial terms to be placed into the polynomial functions for each aerodynamic coefficient. While the white-box nature of a DD model makes this approach appealing, another interesting observation from this work is that the DD is able to more accurately predict the aerodynamic data compared with an LSTM model and an MLP model.

Multi-Fidelity Neural Network Data fusion with neural networks can also be implemented through specialized architectures, as has been done for ASO [88]. He et al. [10] used a modified MLP architecture to incorporate two levels of data fidelity for aerodynamic databases. Inspired by the approach proposed by Meng et al. [89] using multi-fidelity data in conjunction with physics-informed neural networks (PINNs), the authors used a multi-fidelity MLP architecture to fuse data from a low-fidelity CFD solver (Euler) and a high-fidelity solver (Navier–Stokes). The architecture is motivated by the notion that the correlation between high- and low-fidelity data can be given as

$$Y_H = f(X, Y_L) \quad (48)$$

The function $f(\cdot)$ acts as the general mapping from low- to high-fidelity data. This mapping can be decomposed into a linear and a nonlinear part:

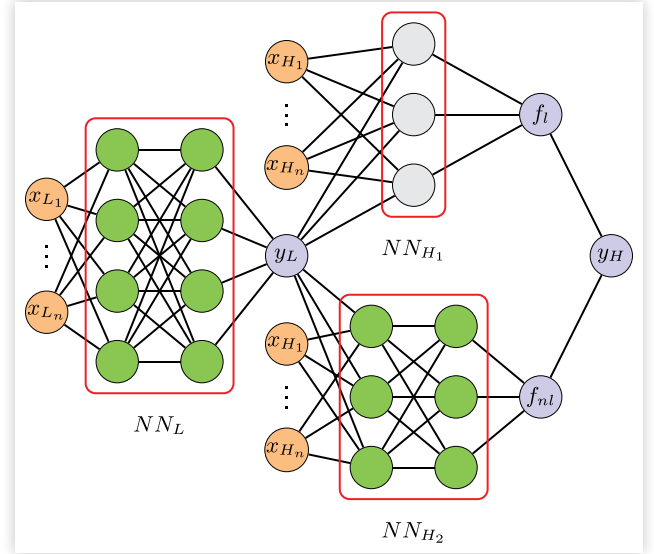
$$f = f_l + f_{nl} \quad (49)$$

Therefore, Equation 48 can be rewritten in terms of these components along with a weighting term, ρ :

$$Y_H = \rho f_l(X, Y_L) + (1 - \rho) f_{nl}(X, Y_L) \quad \rho \in [0, 1] \quad (50)$$

This leads to an architecture comprising three MLPs: the first MLP maps the LF inputs to the LF outputs, and the remaining two MLPs take the LF outputs and map them to HF outputs. Of the remaining MLPs, one uses activation functions within its layers, while the other does not. This is to capture both the linear and nonlinear components of the LF to HF mapping. The architecture for a simplified multi-fidelity neural network is shown in Figure 7. The authors applied this model on a two-dimensional aerofoil dataset with two inputs (AoA, Mach), two outputs (C_L , C_D), 120 LF samples, and 24 HF samples. The model was validated using 96 HF samples, and the test RMSE of the proposed method was found to significantly outperform both VCM (additive scaling function and modeled via kriging) and KOH methods. Further benefits of this method are the ability to simultaneously train on multiple output labels (C_L , C_D , etc.) and more effectively work with large-scale datasets. On the other hand, an inefficient iterative approach was necessary to optimize the model hyperparameters.

FIGURE 7 Architecture of the MFNN used in [10] to implement aerodynamic data fusion. The low-fidelity (NN_L), linear and nonlinear (NN_{H_1} , NN_{H_2}) sub-networks are highlighted. The green nodes indicate the use of a tanh activation function, while the gray nodes indicate the absence of an activation function in the linear sub-network.



Adapted from Ref. [10] and licensed under the Creative Commons 4.0 International Attribution license (<http://creativecommons.org/licenses/by/4.0/>)

Discussion and Concluding Remarks

This section begins by summarizing the methods covered and introducing further approaches drawn from other domains; then it examines how these techniques translate to motorsport aeromaps and highlights the challenges that may arise in that environment.

Summary of Methods

The data fusion methods applied to aerodynamic databases have been categorized into three groups: surrogate-agnostic, kriging-based, and neural network-based. Within surrogate-agnostic methods, scaling functions [7, 62] are straightforward to implement but assume simple multiplicative or additive relationships between datasets that may not capture the complex, nonlinear correlations often present in aerodynamic data. Input vector augmentation [37] is also straightforward to implement and has been shown to be effective when applied to aerodynamic datasets. By adding the output prediction from a model trained on the lower-fidelity data to the higher-fidelity input features, the model can potentially learn more complex relationships between the datasets, rather than being restricted to a constant scaling or additive factor. This is somewhat similar to the NARGP approach introduced by Perdikaris et al. [90] but without being bound to a specific model type. A drawback common to both of these methods is the lack of an uncertainty estimate.

Among the kriging-based methods, the KOH linear autoregressive (AR1) method [67] was the first to apply cokriging to the domain of computer surrogate models, by modeling the higher-fidelity dataset with a scaled low-fidelity dataset and an independent Gaussian process. This model structure is enforced through a specialized covariance matrix. Le Gratiet and Garnier [69] improved on KOH [67] by introducing a recursive structure to reduce computational expense by converting the covariance matrix with all the levels of data into smaller covariance matrices pertaining to each data level. Han et al. proposed a “New Cokriging Method” [38] to simplify and align the cokriging framework with regular kriging; this simpler model structure allows the model hyperparameters to be optimized in one step and also allows for more straightforward implementation of gradient information. Han et al. subsequently proposed hierarchical cokriging [41], which, unlike the AR1 approach, generates plausible uncertainty estimates; a key benefit for adaptive sampling strategies. Unlike the AR1 approach, HK accomplishes data fusion by treating the low-fidelity dataset as the mean or trend function of the kriging model. Scoggins et al. [58] proposed multihierarchy Gaussian processes for applications where the available datasets are not necessarily ordered by fidelity level, and there is not one dataset that can be considered to be the “truth” dataset. The nominal model provides a baseline prediction and uncertainty; the correction terms, constructed using a recursive AR1 framework, provide probabilistic adjustments to obtain a final prediction that ideally more closely matches the ground truth. Yang et al. [59] applied a DBWC fusion method based on pointwise variance optimization and built on a heteroscedastic Gaussian process model as a way to take into account the differences in sample distributions between datasets. With this approach, each level prediction point is weighted in a way that minimizes the overall predictive variance, leading to a model that can handle heterogeneous, sparse datasets more effectively than more traditional approaches such as AR1.

Two primary types of neural network-based data fusion were reviewed. The first type involves fine-tuning, where a network is pre-trained on a low-fidelity dataset and is then retrained on a high-fidelity dataset. During this process, it is possible to freeze a subset of the network’s layers after the pre-training to store some of the high-level patterns from the low-fidelity data. This method has been shown to work effectively with multi-layer perceptrons (MLPs) [24] and dendrite nets (DDs) [11] for aerodynamic database generation.

The other neural network-based approach is the multi-fidelity neural network (MFNN) [10]. This other approach follows the same philosophy as the input vector augmentation and nonlinear autoregressive multi-fidelity Gaussian processes (NARGP) [90] methods, which assume the high-fidelity data can be described as a function of the input data and the low-fidelity data. The difference with MFNNs is that the linear and nonlinear components of the mapping are decomposed into two networks. Neural network-based methods are typically more accommodating to large datasets and can easily

be modified to implement multi-task learning. The drawbacks associated with neural networks for this application are the more involved hyperparameter optimization compared with kriging methods, and the lack of uncertainty estimate; although the latter can be addressed with approaches such as Bayesian neural networks [91].

Related work outside aerodynamic databases offers additional fusion mechanisms with the same goal of exploiting available source information to reduce the number of costly target evaluations and improve model accuracy. Daumé proposed a surrogate-agnostic domain adaptation method [92], which was applied to structural engineering problems by Saida et al. [93], under the name transfer learning Gaussian process regression surrogate model (TL-GPRSM). Data fusion is implemented by stacking the low-fidelity (LF) features and high-fidelity (HF) features vertically, then horizontally stacking a column with just LF data (HF rows are set to zero) and another column with just HF data (LF rows are set to zero). This domain adaptation results in a feature vector of size $\mathbb{R}^{(n_{LF} + n_{HF}) \times 3d}$, where n_{LF} and n_{HF} are the number of LF and HF samples, and d is the number of features. When compared with single-fidelity Gaussian process regression, the TL-GPRSM approach yielded a lower test error for most sample sizes, especially when the number of source domain samples was high. However, the two approaches converged in performance at higher HF sample sizes. One benefit of TL-GPRSM over input vector augmentation is that a LF model does not need to be trained; however, increasing the dimensionality of the feature vector still increases computational cost.

A Bayesian approach can be used to transfer knowledge from a source task to a target task, given some similarity between the two. Prior distributions learned from the source dataset are evaluated using Bayes’ theorem to obtain the MLE for the model parameters. Iapteff et al. [94] applied Bayesian transfer learning to a hydrocracking surrogate model to minimize the number of training samples. In this instance, Bayesian transfer was used to transfer knowledge from a dataset generated with an old catalyst to a dataset generated with a new catalyst. A fully Bayesian kriging framework was adopted, with prior distributions placed over both the covariance function parameters and the trend parameters. The authors used a Gaussian prior for the linear trend and gamma priors for the covariance function parameters. Due to the lack of a closed-form solution for the posterior distributions, a Markov chain Monte Carlo algorithm was used. Transfer learning was applied to two kriging models, one with the linear trend and one with a null trend; both of these methods outperformed the non-transfer equivalents across the entire range of sampling plans. For an RMSE of 0.003, the Bayesian transfer model needed only 25 points instead of the 110 needed by the non-transfer model.

Perdikaris et al. [90] proposed nonlinear autoregressive multi-fidelity Gaussian processes (NARGP) to consolidate multi-fidelity datasets with more complex nonlinear correlations. The method relies on the recursive framework introduced by [69] but uses a different generalization

of the autoregressive multi-fidelity scheme that is found in the AR1 method. This method is similar to that of Ghoreyshi et al. [37], where the HF prediction is treated as a function of both the features and the prediction from a lower-fidelity model: $y_t(\mathbf{x}) = g_t(\mathbf{x}, \hat{y}_{t-1}(\mathbf{x}))$; the difference being that NARGP uses a specialized composite kernel. This formulation comprises a composition of two GP priors, which is, in effect, a deep GP, as introduced by [95]. However, due to its recursive nature, the NARGP method also relies on the assumption of nested datasets, as with other cokriging methods. The authors validated the NARGP approach by comparing it with standard GPR and AR1 models for mixed-convection flow modeling, using the Nusselt number as the output and several flow-related variables as inputs. The NARGP model performed consistently better than the AR1 and single-fidelity GPR models, suggesting that its capacity to learn more complex relationships between datasets provides a practical benefit in certain cases. Furthermore, the AR1 model only significantly outperformed the GPR model at the lowest and highest sampling plan sizes, while the NARGP was consistently better across the whole range of sampling sizes. In the same vein, Cutajar et al. [96] extended the concept of deep Gaussian processes (DGPs) to multi-fidelity data (MF-DGP). By treating layers of a DGP as fidelity levels, the model can capture nonlinear relationships across datasets. MF-DGP was applied to several synthetic datasets and compared with other multi-fidelity models, including NARGP and AR1, where it outperformed the other models in the majority of datasets.

Key Findings

The central finding of this review is that classical multi-fidelity hierarchies may not translate directly to motorsport aeromaps. In the aerospace literature, LF datasets are typically generated using simplified numerical solvers (panel methods, Euler solvers) that, despite their approximations, capture the essential physics of the flow. For F1 geometries, however, the complex viscous flow structures are unlikely to be adequately represented by such solvers. Consequently, conventional LF data sources may exhibit weak or misleading correlations with higher-fidelity wind tunnel or CFD data, limiting the effectiveness of standard multi-fidelity approaches.

This observation shifts the emphasis from *solver-fidelity fusion* toward a *transfer learning paradigm*, in which legacy aeromap data from a previous geometry serves as the source dataset, and the updated geometry defines the target. This reframing aligns with approaches demonstrated by Ghoreyshi et al. [37] and Mohammadi-Amin et al. [46], who used data fusion to update aerodynamic databases following geometry modifications. However, it also introduces the risk of negative transfer if the two geometries, and consequently the datasets, are too different.

Beyond this central finding, the following observations emerge from the literature:

- **Kriging-based** methods dominate the aerodynamic database literature and provide native uncertainty quantification, which can be useful for validation and adaptive sampling. However, most reported applications employ models that assume approximately linear cross-dataset relationships (AR1, NCM, HK). Methods capable of capturing nonlinear correlations, such as NARGP and MF-DGP, remain underexplored in this context.
- **Neural network** architectures offer superior scalability and native multi-task learning, enabling joint modeling of correlated outputs (e.g., C_L , C_D). Architectures such as MFNNs have demonstrated strong performance in aerospace applications, outperforming both scaling functions and kriging-based methods in some comparative studies [10].
- **Surrogate-agnostic** methods (scaling functions, input vector augmentation) are straightforward to implement but capture only limited cross-dataset relationships and do not provide predictive uncertainty estimates. These characteristics make them less suitable than kriging- or neural network-based alternatives for F1 aeromap modeling, where uncertainty quantification supports both model validation and adaptive sampling strategies.

Method Selection Considerations

The choice of data fusion technique should be guided by four principal factors:

1. **Nature of the cross-dataset relationship:** Predominantly linear correlations are well served by AR1, NCM, or HK. Strongly nonlinear correlations benefit from pre-trained neural networks, MFNNs, or NARGP.
2. **Scale of the source dataset:** Neural networks scale readily to large datasets, whereas kriging-based methods require modifications such as sparse approximations [97] or localized approaches to remain computationally tractable.
3. **Multi-output requirements:** When multiple correlated outputs must be modeled jointly, neural networks handle multi-task learning natively. Kriging requires multi-output GP formulations [98], which adds complexity.
4. **Uncertainty quantification:** Kriging methods provide uncertainty estimates directly. Neural networks require additional techniques such as Monte Carlo dropout [99], ensembles, or Bayesian architectures [91].

For F1 aeromaps, two operational scenarios warrant consideration:

- **In-season development** involves incremental geometry changes: modified front/rear wing or revised floor details, for example, where the overall vehicle geometry remains relatively consistent. In this setting, source and target aeromaps are likely to share substantial correlation structure, and data

fusion should offer clear benefits provided the incremental changes do not fundamentally alter the dominant flow features.

- **Regulation-driven changes** present a more demanding test case. Major regulatory changes can produce step changes in vehicle geometry and flow topology. Under such conditions, legacy aeromaps may exhibit weak or even misleading correlations with the new configuration, increasing the risk of negative transfer. Whether data fusion remains beneficial in this scenario or whether a clean-slate single-fidelity approach is preferable remains an open question.

In both scenarios, validation is key. Comparing multi-fidelity model performance against a single-fidelity baseline using a validation dataset or LOOCV provides a safeguard: if the multi-fidelity model does not outperform the single-fidelity model, the former should be disregarded or down-weighted.

Proposed Research Directions

To address the gaps identified in this review, the following research directions are proposed:

1. **Comparative evaluation of fusion strategies for motorsport aeromaps.** This should benchmark neural network-based approaches (pre-training with fine-tuning, MFNNs) against kriging-based approaches (AR1, HK, NARGP) using representative F1 aeromap data. Evaluation criteria should include predictive accuracy, uncertainty quality, multi-output performance, computational cost, and, most importantly, the number of target samples required to achieve a specified accuracy threshold.
2. **Quantification of source-target similarity requirements.** Establishing metrics that predict when data fusion will be beneficial and when it will degrade performance would help guide practitioners. This investigation should span both in-season and regulation-change scenarios.
3. **Development of negative transfer mitigation strategies.** Approaches such as ensemble methods should be evaluated in the motorsport context. The goal is to ensure that data fusion either improves or, at worst, matches single-fidelity performance across the full range of source-target similarity conditions encountered in F1 development.

Conclusion

This paper has presented one of the first motorsport-focused reviews of data fusion methods for aerodynamic database generation. The surveyed approaches were categorized into three groups: surrogate-agnostic, kriging-based, and neural network-based. Additional fusion mechanisms from adjacent domains, including domain

adaptation, Bayesian transfer learning, and nonlinear autoregressive Gaussian processes, were also reviewed to identify techniques that may address limitations of established methods applied to aerodynamic databases. The principal contribution is the identification of a paradigm shift required for F1 applications: from multi-fidelity fusion toward geometry-to-geometry transfer learning, with legacy aeromap data serving as the source dataset. This shift introduces challenges, particularly negative transfer risk, that are not prominent in the aerospace literature but are central to effective deployment in motorsport. The proposed research directions aim to address these challenges and establish a foundation for data fusion strategies that can reduce data generation cost while maintaining the predictive accuracy essential for effective aerodynamic development.

Acknowledgements

This work was funded and supported by Formula One Research, Engineering and Development Limited, via the Formula 1/Amazon Web Services Joint Technology Innovation Fund, for the project: "Development of a Strategy to Generate a Formula 1 Car Aeromap From a Limited Number of Test Points". We would like to acknowledge Craig Wilson for serving as the industrial supervisor and providing valuable insights into motorsport vehicle dynamics and aerodynamics, as well as Simon Dodman and Pablo Hermoso Moreno. We would also like to thank Pat Symonds for his role as industrial supervisor in the initial stages of the project.

Contact Information

thomas.j.ongley@cranfield.ac.uk

Definitions, Acronyms, Abbreviations

ANN - Artificial Neural Network
AR1 - First-Order Autoregressive (Cokriging)
DBWC - Distribution-Based Weight Control
DD - Dendrite Net
DoE - Design of Experiments
GCK - Generalized Hierarchical Cokriging
GPR - Gaussian Process Regression
HF - High-Fidelity
HGP - Heteroscedastic Gaussian Process
HK - Hierarchical Kriging
IHK - Improved Hierarchical Kriging
KOH - Kennedy and O'Hagan Cokriging Model

LF - Low-Fidelity

MFNN - Multi-Fidelity Neural Network

MHGPR - Multihierarchy Gaussian Process Regression

MLP - Multilayer Perceptron

NCM - New Cokriging Model

RMSE - Root Mean Squared Error

References

1. Fédération Internationale de l'Automobile, "2025 Formula One Sporting Regulations - Issue 4," February 2025.
2. Fédération Internationale de l'Automobile, "2025 Formula 1 Technical Regulations - Issue 3," July 2025.
3. Garud, S.S., Karimi, I.A., and Kraft, M., "Design of Computer Experiments: A Review," *Computers & Chemical Engineering* 106 (2017): 71-95.
4. Rataczak, J.A., Boyd, I.D., and McMahon, J.W., "Surrogate Models for Hypersonic Aerothermodynamics and Aerodynamics Using Gaussian Process Regression," in *AIAA SCITECH 2024 Forum*, Orlando, FL, American Institute of Aeronautics and Astronautics, January 2024.
5. Quinlan, K.R., Movva, J., Stein, E.V., and Kupresanin, A., "Leveraging Multi-Fidelity Aerodynamic Databasing to Efficiently Represent a Hypersonic Design Space," in *ASCEND 2021*, Las Vegas, NV and Virtual, American Institute of Aeronautics and Astronautics, November 2021.
6. Kuya, Y., Takeda, K., Zhang, X., and Forrester, A.I.J., "Multifidelity Surrogate Modeling of Experimental and Computational Aerodynamic Data Sets," *AIAA Journal* 49 (2011): 289-298.
7. Tang, C., Gee, K., and Lawrence, S., "Generation of Aerodynamic Data Using a Design of Experiment and Data Fusion Approach," in *43rd AIAA Aerospace Sciences Meeting and Exhibit*, Reno, NV, American Institute of Aeronautics and Astronautics, January 2005.
8. Ghoreyshi, M., Badcock, K., and Woodgate, M., "Integration of Multi-Fidelity Methods for Generating an Aerodynamic Model for Flight Simulation," in *46th AIAA Aerospace Sciences Meeting and Exhibit*, Reno, NV, American Institute of Aeronautics and Astronautics, January 2008.
9. Tyan, M., Kim, M., Pham, V., Choi, C.K. et al., "Development of Advanced Aerodynamic Data Fusion Techniques for Flight Simulation Database Construction," in *2018 Modeling and Simulation Technologies Conference*, Atlanta, GA, American Institute of Aeronautics and Astronautics, June 2018.
10. He, L., Qian, W., Zhao, T., and Wang, Q., "Multi-Fidelity Aerodynamic Data Fusion with a Deep Neural Network Modeling Method," *Entropy* 22 (2020): 1022.
11. Ni, Z., Chen, S., Zhu, Z., and Wang, W., "Aerodynamic Modeling of a Missile Combining Transfer Learning and Dendritic Net," *Journal of Aircraft* 61 (2024): 1535-1549.
12. Katz, J., "Aerodynamics in Motorsports," *Proceedings of the Institution of Mechanical Engineers, Part P* 235 (2021): 324-338.
13. Gadola, M., Candelpergher, A., and Adami, R., "The Impact of Non-Linear Aerodynamics on Racecar Behaviour and Lap Time Simulation," *SAE Technical Paper 2002-01-3332* (2002), doi:<https://doi.org/10.4271/2002-01-3332>.
14. McKay, M.D., Beckman, R.J., and Conover, W.J., "A Comparison of Three Methods for Selecting Values of Input Variables in the Analysis of Output from a Computer Code," *Technometrics* 21 (1979): 239.
15. Owen, A.B., "Orthogonal Arrays for Computer Experiments, Integration and Visualization," *Statistica Sinica* 2, no. 2 (1992): 439-452.
16. Liu, H., Cai, J., and Ong, Y.-S., "An Adaptive Sampling Approach for Kriging Metamodeling by Maximizing Expected Prediction Error," *Computers & Chemical Engineering* 106 (2017): 171-182.
17. Toet, W., "Aerodynamics and Aerodynamic Research in Formula 1," *The Aeronautical Journal* 117 (2013): 1-26.
18. Bushnell, D.M., "SCALING: Wind Tunnel to Flight," *Annual Review of Fluid Mechanics* 38 (2006): 111-128.
19. Ashton, N., Sachs, S., Foti, L., and Eberhardt, S., "Towards High-Fidelity CFD on the Cloud for the Automotive and Motorsport Sectors," *SAE Technical Paper 2020-01-0665* (2020), doi:<https://doi.org/10.4271/2020-01-0665>.
20. Bush, R.H., Chyczewski, T.S., Duraisamy, K., Eisfeld, B. et al., "Recommendations for Future Efforts in RANS Modeling and Simulation," in *AIAA Scitech 2019 Forum*, San Diego, CA, American Institute of Aeronautics and Astronautics, January 2019.
21. Heinz, S., "A Review of Hybrid RANS-LES Methods for Turbulent Flows: Concepts and Applications," *Progress in Aerospace Sciences* 114 (2020): 100597.
22. Jackson, B. and Cruz, C., "Preliminary Subsonic Aerodynamic Model for Simulation Studies of the HL-20 Lifting Body.pdf," Technical Memorandum 19920021931, NASA, January 1992.
23. Lee, D., Van Nguyen, N., Tyan, M., Chun, H.-G. et al., "Enhanced Multi-Fidelity Model for Flight Simulation Using Global Exploration and the Kriging Method," *Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering* 231 (2017): 606-620.
24. Rajkumar, T. and Bardina, J., "Prediction of Aerodynamic Coefficients Using Neural Networks for Sparse Data," in *Proceedings of the Fifteenth International Florida Artificial Intelligence Research Society Conference (FLAIRS 2002)*, Pensacola Beach, FL, 242-246, AAAI, 2002.
25. Wang, Q., Qian, W., and He, K., "Unsteady Aerodynamic Modeling at High Angles of Attack Using Support Vector Machines," *Chinese Journal of Aeronautics* 28 (2015): 659-668.
26. Sacks, J., Welch, W.J., Mitchell, T.J., and Wynn, H.P., "Design and Analysis of Computer Experiments," *Statistical Science* 4 (1989): 409-423.

27. O'Hagan, A., "Curve Fitting and Optimal Design for Prediction," *Journal of the Royal Statistical Society Series B: Statistical Methodology* 40 (1978): 1-24.
28. Ollar, J., Mortished, C., Jones, R., Siens, J. et al., "Gradient Based Hyper-Parameter Optimisation for Well Conditioned Kriging Metamodels," *Structural and Multidisciplinary Optimization* 55 (2017): 2029-2044.
29. Muyskens, A., Priest, B., Goumiri, I., and Schneider, M., "MuyGPs: Scalable Gaussian Process Hyperparameter Estimation Using Local Cross-Validation," arXiv:2104.14581 [stat], April 2021.
30. Goodfellow, I., Bengio, Y., and Courville, A., *Deep Learning* (Cambridge, MA: MIT Press, 2016).
31. Bishop, C.M., *Pattern Recognition and Machine Learning. Information Science and Statistics* (New York: Springer, 2006).
32. Apicella, A., Donnarumma, F., Isgro, F., and Prevete, R., "A Survey on Modern Trainable Activation Functions," *Neural Networks* 138 (2021): 14-32.
33. Duchi, J., Hazan, E., and Singer, Y., "Adaptive Subgradient Methods for Online Learning and Stochastic Optimization," *Journal of Machine Learning Research* 12 (2011): 2121-2159.
34. Hinton, G., Srivastava, N., and Swersky, K., "Lecture 6d: A Separate, Adaptive Learning Rate for Each Connection," 2012.
35. Kingma, D.P. and Ba, J., "Adam: A Method for Stochastic Optimization," arXiv:1412.6980 [cs], January 2017.
36. Tang, C., Gee, K., and Hawke, V., "Development of Intelligent Agents for the Generation of Aerodynamic Data," in *41st Aerospace Sciences Meeting and Exhibit*, Reno, NV, American Institute of Aeronautics and Astronautics, January 2003.
37. Ghoreyshi, M., Badcock, K.J., and Woodgate, M.A., "Accelerating the Numerical Generation of Aerodynamic Models for Flight Simulation," *Journal of Aircraft* 46 (2009): 972-980.
38. Han, Z.-H., Zimmermann, R., and Goretz, S., "A New Cokriging Method for Variable-Fidelity Surrogate Modeling of Aerodynamic Data," in *48th AIAA Aerospace Sciences Meeting Including the New Horizons Forum and Aerospace Exposition*, Orlando, FL, American Institute of Aeronautics and Astronautics, January 2010.
39. Yamazaki, W. and Mavriplis, D., "Derivative-Enhanced Variable Fidelity Surrogate Modeling for Aerodynamic Functions," in *49th AIAA Aerospace Sciences Meeting Including the New Horizons Forum and Aerospace Exposition*, Orlando, FL, American Institute of Aeronautics and Astronautics, January 2011.
40. Da Ronch, A., Ghoreyshi, M., and Badcock, K.J., "On the Generation of Flight Dynamics Aerodynamic Tables by Computational Fluid Dynamics," *Progress in Aerospace Sciences* 47 (2011): 597-620.
41. Han, Z.-H. and Görtz, S., "Hierarchical Kriging Model for Variable-Fidelity Surrogate Modeling," *AIAA Journal* 50 (2012): 1885-1896.
42. Cristofaro, M., Wang, Y., and Da Ronch, A., "Towards Computational Flight Dynamics of a Passenger Aircraft," in *Proceedings of the 29th Congress of the International Council of the Aeronautical Sciences*, St. Petersburg, Russia, September 2014.
43. Tancred, J. and Rumpfkeil, M.P., "Aerodynamic Response Quantification of Complex Hypersonic Configurations Using Variable Fidelity Surrogate Modeling," in *53rd AIAA Aerospace Sciences Meeting*, Kissimmee, FL, American Institute of Aeronautics and Astronautics, January 2015.
44. Rumpfkeil, M.P. and Beran, P., "Construction of Multi-Fidelity Surrogate Models for Aerodynamic Databases," in *Proceedings of the Ninth International Conference on Computational Fluid Dynamics (ICCFD9)*, Istanbul, Turkey, July 2016.
45. He, L., Zhou, Y., and Qian, W., "Aerodynamic Data Fusion with a Multi-Fidelity Surrogate Modeling Method," in *Proceedings of 7th European Conference for Aeronautics and Space Sciences (EUCASS 2017)*, Milano, Italy, July 3-6, 2017, 10 pages.
46. Mohammadi-Amin, M., Entezari, M.M., and Alikhani, A., "An Efficient Surrogate-Based Framework for Aerodynamic Database Development of Manned Reentry Vehicles," *Advances in Space Research* 62 (2018): 997-1014.
47. Kaya, H., Tiftikçi, H., Kutluay, and Sakarya, E., "Generation of Surrogate-Based Aerodynamic Model of an UCAV Configuration Using an Adaptive Co-Kriging Method," *Aerospace Science and Technology* 95 (2019): 105511.
48. Kim, M., Tyan, M., Juliawan, N., Kim, S. et al., "Data Fusion Method for Constructing Fighter Aircraft Aerodynamic Database Using Variable Fidelity and Space Mapping Techniques," in *Proceedings of the 8th European Conference for Aeronautics and Space Sciences*, Madrid, Spain, July 1-4, 2019, 10 pages.
49. Mukhopadhyaya, J., Whitehead, B.T., Quindlen, J.F., Alonso, J.J. et al., "Multi-Fidelity Modeling of Probabilistic Aerodynamic Databases for Use in Aerospace Engineering," *International Journal for Uncertainty Quantification* 10, no. 5 (2020): 425-447.
50. Zhang, M., Bartoli, N., Jungo, A., Lammen, W. et al., "Enhancing the Handling Qualities Analysis by Collaborative Aerodynamics Surrogate Modelling and Aero-Data Fusion," *Progress in Aerospace Sciences* 119 (2020): 100647.
51. Arenzana, R.C., López-Lopera, A.F., Mouton, S., Bartoli, N. et al., "Multi-Fidelity Gaussian Process Model for CFD and Wind Tunnel Data Fusion," in *Proceedings of International Conference on Multidisciplinary Design Optimization of Aerospace Systems (Aerobest 2021)*, Lisbon, Portugal, July 2021.
52. Lee, S., Kang, S., and Lee, K., "Development of a Gaussian Process Modeling Application for the Prediction of Missile Aerodynamic Coefficients," *Journal of Mechanical Science and Technology* 35 (2021): 987-997.
53. Burchett, B.T., Vasile, J.D., and Bryson, J., "Combining Sparse and Dense Databases to Form a Robust Aerodynamic Model for a Long-Range, High-Speed Projectile," in *AIAA SCITECH 2022 Forum*, San Diego, CA

- and Virtual, American Institute of Aeronautics and Astronautics, January 2022.
54. Peng, X., Kou, J., and Zhang, W., "Multi-Fidelity Nonlinear Unsteady Aerodynamic Modeling and Uncertainty Estimation Based on Hierarchical Kriging," *Applied Mathematical Modelling* 122 (2023): 1-21.
 55. Pham, V., Tyan, M., Nguyen, T.A., and Lee, J.-W., "Extended Hierarchical Kriging Method for Aerodynamic Model Generation Incorporating Multiple Low-Fidelity Datasets," *Aerospace* 11 (2023): 6.
 56. Quinlan, K.R., Movva, J., Burchett, B.T., Vasile, J.D. et al., "Evaluation of Multi-Fidelity Approaches with Active Learning for Automated Database Generation," in *AIAA AVIATION 2023 Forum*, San Diego, CA and Online, American Institute of Aeronautics and Astronautics, June 2023.
 57. Burchett, B., "Multi-Fidelity Data Fusion for Aerodynamic Prediction Including Parametric Uncertainty," Tech. Rep., DEVCOM Army Research Laboratory, Aberdeen Proving Ground, MD, October 2023.
 58. Scoggins, J.B., Wignall, T.J., Nakamura-Zimmerer, T., and Bibb, K.L., "Multihierarchy Gaussian Process Models for Probabilistic Aerodynamic Databases Using Uncertain Nominal and Off-Nominal Configuration Data," in *AIAA SCITECH 2023 Forum*, National Harbor, MD and Online, American Institute of Aeronautics and Astronautics, January 2023.
 59. Yang, B., Chen, B., Liu, Y., and Chen, J., "Gaussian Process Fusion Method for Multi-Fidelity Data with Heterogeneity Distribution in Aerospace Vehicle Flight Dynamics," *Engineering Applications of Artificial Intelligence* 138 (2024): 109228.
 60. Xu, C., Han, Z., Zhang, K., Zeng, H. et al., "A Novel Multi-Fidelity Cokriging Model Assisted by Multiple Non-Hierarchical Low-Fidelity Datasets," *Structural and Multidisciplinary Optimization* 67 (2024): 52.
 61. Li, J., Du, X., and Martins, J.R., "Machine Learning in Aerodynamic Shape Optimization," *Progress in Aerospace Sciences* 134 (2022): 100849.
 62. Hutchison, M., Huang, X., Mason, W., Haftka, R. et al., "Variable-Complexity Aerodynamic-Structural Design of a High-Speed Civil Transport Wing," in *4th Symposium on Multidisciplinary Analysis and Optimization*, Cleveland, OH, American Institute of Aeronautics and Astronautics, September 1992.
 63. Haftka, R.T., "Combining Global and Local Approximations," *AIAA Journal* 29 (1991): 1523-1525.
 64. Lewis, R. and Nash, S., "A Multigrid Approach to the Optimization of Systems Governed by Differential Equations," in *8th Symposium on Multidisciplinary Analysis and Optimization*, Long Beach, CA, American Institute of Aeronautics and Astronautics, September 2000.
 65. Journel, A.G. and Huijbregts, C.J., *Mining Geostatistics* (New York: Academic Press, 1978).
 66. Myers, D.E., "Matrix Formulation of Co-Kriging," *Journal of the International Association for Mathematical Geology* 14 (1982): 249-257.
 67. Kennedy, M., "Predicting the Output from a Complex Computer Code When Fast Approximations Are Available," *Biometrika* 87 (2000): 1-13.
 68. Brevault, L., Balesdent, M., and Hebbal, A., "Overview of Gaussian Process Based Multi-Fidelity Techniques with Variable Relationship between Fidelities, Application to Aerospace Systems," *Aerospace Science and Technology* 107 (2020): 106339.
 69. Le Gratiet, L. and Garnier, J., "Recursive Co-Kriging Model for Design of Computer Experiments with Multiple Levels of Fidelity," *International Journal for Uncertainty Quantification* 4, no. 5 (2014): 365-386.
 70. Goovaerts, P., *Geostatistics for Natural Resources Evaluation* (Oxford: Oxford University Press, 1997).
 71. Yamazaki, W., Rumpfkeil, M., and Mavriplis, D., "Design Optimization Utilizing Gradient/Hessian Enhanced Surrogate Model," in *28th AIAA Applied Aerodynamics Conference, Fluid Dynamics and Co-located Conferences*, Chicago, IL, American Institute of Aeronautics and Astronautics, June 2010.
 72. Zhou, Q., Wu, Y., Guo, Z., Hu, J. et al., "A Generalized Hierarchical Co-Kriging Model for Multi-Fidelity Data Fusion," *Structural and Multidisciplinary Optimization* 62 (2020): 1885-1904.
 73. Kou, J. and Zhang, W., "Multi-Kernel Neural Networks for Nonlinear Unsteady Aerodynamic Reduced-Order Modeling," *Aerospace Science and Technology* 67 (2017): 309-326.
 74. Xu, C.-Z., Han, Z.-H., Zan, B.-W., Zhang, K.-S. et al., "Expert's Experience-Informed Hierarchical Kriging Method for Aerodynamic Data Modeling," *Engineering Applications of Artificial Intelligence* 133 (2024): 108490.
 75. Cheng, M., Jiang, P., Hu, J., Shu, L. et al., "A Multi-Fidelity Surrogate Modeling Method Based on Variance-Weighted Sum for the Fusion of Multiple Non-Hierarchical Low-Fidelity Data," *Structural and Multidisciplinary Optimization* 64 (2021): 3797-3818.
 76. Pan, S.J. and Yang, Q., "A Survey on Transfer Learning," *IEEE Transactions on Knowledge and Data Engineering* 22 (2010): 1345-1359.
 77. Yan, X., Zhu, J., Kuang, M., and Wang, X., "Aerodynamic Shape Optimization Using a Novel Optimizer Based on Machine Learning Techniques," *Aerospace Science and Technology* 86 (2019): 826-835.
 78. Liao, P., Song, W., Du, P., and Zhao, H., "Multi-Fidelity Convolutional Neural Network Surrogate Model for Aerodynamic Optimization Based on Transfer Learning," *Physics of Fluids* 33 (2021): 127121.
 79. Norgaard, M., Jorgensen, C.C., and Ross, J.C., "Neural Network Prediction of New Aircraft Design Coefficients," Technical Memorandum 112197, NASA, Ames Research Center, Mountain View, CA, May 1997.
 80. Rai, M. and Madavan, N., "Aerodynamic Design Using Neural Networks," in *7th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization*, St. Louis, MO, American Institute of Aeronautics and Astronautics, September 1998.

81. Lo, C., Zhao, J., and DeLoach, R., "Application of Neural Networks to Wind Tunnel Data Response Surface Methods," in *21st Aerodynamic Measurement Technology and Ground Testing Conference*, Denver, CO, American Institute of Aeronautics and Astronautics, June 2000.
82. Wu, M.-Y., He, X.-J., Sun, X.-H., Tong, T.-S. et al., "Efficient Aerodynamic Shape Optimization Using Transfer Learning Based Multi-Fidelity Deep Neural Network," *Physics of Fluids* 36 (2024): 116109.
83. Leng, J.-X., Feng, Y., Huang, W., Shen, Y. et al., "Variable-Fidelity Surrogate Model Based on Transfer Learning and Its Application in Multidisciplinary Design Optimization of Aircraft," *Physics of Fluids* 36 (2024): 017131.
84. Smith, J.S., Nebgen, B.T., Zubatyuk, R., Lubbers, N. et al., "Approaching Coupled Cluster Accuracy with a General-Purpose Neural Network Potential through Transfer Learning," *Nature Communications* 10 (2019): 2903.
85. Shen, S., Sadoughi, M., Li, M., Wang, Z. et al., "Deep Convolutional Neural Networks with Ensemble Learning and Transfer Learning for Capacity Estimation of Lithium-Ion Batteries," *Applied Energy* 260 (2020): 114296.
86. Liu, G. and Wang, J., "Dendrite Net: A White-Box Module for Classification, Regression, and System Identification," *IEEE Transactions on Cybernetics* 52 (2022): 13774-13787.
87. Huang, J., Liu, K., Wang, Q., Qiao, W. et al., "Control-Oriented Aerodynamic Modeling with Automatic White-Box Structure Based on Dendrite Net," in *2023 IEEE 13th International Conference on CYBER Technology in Automation, Control, and Intelligent Systems (CYBER)*, Qinhuangdao, China, 921-926, IEEE, July 2023.
88. Zhang, X., Xie, F., Ji, T., Zhu, Z. et al., "Multi-Fidelity Deep Neural Network Surrogate Model for Aerodynamic Shape Optimization," *Computer Methods in Applied Mechanics and Engineering* 373 (2021): 113485.
89. Meng, X. and Karniadakis, G.E., "A Composite Neural Network that Learns from Multi-Fidelity Data: Application to Function Approximation and Inverse PDE Problems," *Journal of Computational Physics* 401 (2020): 109020.
90. Perdikaris, P., Raissi, M., Damianou, A., Lawrence, N.D. et al., "Nonlinear Information Fusion Algorithms for Data-Efficient Multi-Fidelity Modelling," *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences* 473 (2017): 20160751.
91. Olivier, A., Shields, M.D., and Graham-Brady, L., "Bayesian Neural Networks for Uncertainty Quantification in Data-Driven Materials Modeling," *Computer Methods in Applied Mechanics and Engineering* 386 (2021): 114079.
92. Daumé, H. III, "Frustratingly Easy Domain Adaptation," arXiv:0907.1815 [cs], July 2009.
93. Saida, T. and Nishio, M., "Transfer Learning Gaussian Process Regression Surrogate Model with Explainability for Structural Reliability Analysis under Variation in Uncertainties," *Computers & Structures* 281 (2023): 107014.
94. Iapteff, L., Jacques, J., Rolland, M., and Celse, B., "Reducing the Number of Experiments Required for Modelling the Hydrocracking Process with Kriging through Bayesian Transfer Learning," *Journal of the Royal Statistical Society Series C: Applied Statistics* 70 (2021): 1344-1364.
95. Damianou, A. and Lawrence, N., "Deep Gaussian Processes," in *Proceedings of the Sixteenth International Conference on Artificial Intelligence and Statistics*, Scottsdale, AZ, vol. 31, 207-215, 2013.
96. Cutajar, K., Pullin, M., Damianou, A., Lawrence, N. et al., "Deep Gaussian Processes for Multi-fidelity Modeling," arXiv:1903.07320 [stat], March 2019.
97. Lawrence, N.D., Seeger, M., and Herbrich, R., "Fast Sparse Gaussian Process Methods: The Informative Vector Machine," in *NeurIPS Proceedings*, Vancouver, BC, Canada, 2002.
98. Bonilla, E.V., Chai, K., and Williams, C., "Multi-task Gaussian Process Prediction," in Platt, J., Koller, D., Singer, Y., and Roweis, S. (eds), *Advances in Neural Information Processing Systems*, vol. 20 (Red Hook, NY: Curran Associates, Inc., 2007).
99. Gal, Y. and Ghahramani, Z., "Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning," in *Proceedings of the 33rd International Conference on Machine Learning*, New York, 1050-1059, PMLR, June 2016.