

See discussions, stats, and author profiles for this publication at: <https://www.researchgate.net/publication/339831919>

# Scalable gradient-enhanced artificial neural networks for airfoil shape design in the subsonic and transonic regimes

Article in Structural and Multidisciplinary Optimization · April 2020

DOI: 10.1007/s00158-020-02488-5

CITATIONS

8

READS

562

## 3 authors:



Mohamed Amine Bouhlel

University of Michigan

14 PUBLICATIONS 343 CITATIONS

[SEE PROFILE](#)



Sicheng He

University of Michigan

14 PUBLICATIONS 69 CITATIONS

[SEE PROFILE](#)



Joaquim R. R. A. Martins

University of Michigan

305 PUBLICATIONS 9,207 CITATIONS

[SEE PROFILE](#)

## Some of the authors of this publication are also working on these related projects:



Project

Nonlocal, Nonparallel Flow Stability Analysis Transition to Turbulence Prediction over Aeronautical Applications [View project](#)



Project

Passive Aeroelastic Tailored Wing [View project](#)

# Scalable gradient-enhanced artificial neural networks for airfoil shape design in the subsonic and transonic regimes

Mohamed Amine Bouhlel<sup>a</sup>, Sicheng He<sup>a</sup>, Joaquim R. R. A. Martins<sup>a</sup>

<sup>a</sup>*University of Michigan, Department of Aerospace Engineering, Ann Arbor, MI, USA*

---

## Abstract

Airfoil shape design is one of the most fundamental elements in aircraft design. Existing airfoil design tools require at least a few minutes to analyze a new shape and hours to perform shape optimization. To drastically reduce the computational time of both analysis and design optimization, we use machine learning to create a model of a wide range of possible airfoils at a range of flight conditions, making it possible to perform airfoil design optimization in a few seconds. The machine learning consists of gradient-enhanced artificial neural networks where the gradient information is phased in gradually. This new gradient-enhanced artificial neural network approach is trained to model the aerodynamic force coefficients of airfoils in both subsonic and transonic regimes. The aerodynamics is modeled with Reynolds-averaged Navier–Stokes (RANS) based computational fluid dynamics (CFD). The proposed approach outperforms an existing airfoil model that uses a mixture of experts technique combined with a gradient-based Kriging surrogate model. The approach yields to similar airfoil shape optimization solutions than high-fidelity CFD optimization solutions with a difference of 0.01 count and 0.12 count for  $C_d$  in subsonic and transonic regimes, respectively. Airfoil optimization problems are solved in a few seconds (instead of hours using CFD-based optimization), making the design process much more interactive, as demonstrated in the Webfoil airfoil design optimization tool.

*Keywords:* Surrogate Modeling, Artificial Neural Networks, Gradient-Enhanced Modeling, Airfoils

---

## 1. Introduction

Surrogate modeling methods are a powerful tool that have applications in engineering analysis and design tasks. If an outcome of interest is time-consuming to evaluate using numerical simulations, a fast surrogate model that approximates the output can be used instead. Many types of surrogate models exist in the literature, such as kriging [51, 48, 9, 8, 32, 17], radial basis functions (RBF) [46, 13, 24], and artificial neural networks (ANN) [26, 23]. Gradient-enhanced surrogate modeling methods use the gradient information to improve the accuracy of the model [34, 7]. These methods have been adapted to many surrogate model types, such as kriging [7, 5, 16] and RBF [57, 31].

---

*Email addresses:* mbouhlel@umich.edu (Mohamed Amine Bouhlel), hschsc@umich.edu (Sicheng He), jrram@umich.edu (Joaquim R. R. A. Martins)

10 Laurent et al. [34] provide an overview of existing gradient-enhanced modeling and their  
11 applications. However, most of the gradient-enhanced surrogate modeling methods do  
12 not scale well in number of samples (more than tens of thousands), which limits the  
13 achievable accuracy, especially for higher dimensional design spaces.

14 ANN have become increasingly popular due to the increased availability of training  
15 data and computing capability. One of the advantages of ANN is their scalability with the  
16 number of samples [21]. ANN are a computational model inspired by the way biological  
17 neural networks in the human brain process information and were originally invented by  
18 McCulloch and Pitts [43]. Goodfellow et al. [21] describes the evolution of ANN, which  
19 have had a history that has alternated between progress and stagnation. Today, ANN  
20 are used in many practical applications and active research topics such as autonomous  
21 vehicles [2], computer vision [3], and speech recognition [30].

22 ANN have been also used in aerospace applications. Rai and Madavan [47] used ANN  
23 and a polynomial model to approximate the pressure distribution of a turbine blade in the  
24 subsonic regime involving 15 design variables. They adapted the neural net dynamically  
25 during the optimization, which made it time-consuming. To manage the trade-off between  
26 accuracy and efficiency, they based the surrogate model on Euler data at the initial stage  
27 of optimization and switched to RANS data in the final stage. Papila et al. [45] optimized  
28 the performance of a supersonic turbine using radial basis neural networks combined with  
29 polynomial models using data generated by an unsteady RANS solver. The ANN was  
30 trained with the CFD data and was then used to generate more data points for the  
31 polynomial model. Zhang et al. [56] analyzed the image of an airfoil and estimated its  
32 lift coefficient using convolutional neural networks with training data generated from a  
33 RANS solver. Xu et al. [53] used ANN as a surrogate relating an airfoil geometry and its  
34 buffeting magnitude, frequency and time-average aerodynamic load coefficients based on  
35 unsteady RANS simulations.

36 ANN have been extended to multi-information sources modeling. However, there has  
37 been little research on ANN that use gradient information to solve engineering problems.  
38 Giannakoglou et al. [19] analyzed a number of 2D and 3D aerodynamic shape design  
39 problems using gradient-enhanced ANN. Their error metric was a combination of function  
40 error and gradient error. They used an Euler equation model and thus viscous effects  
41 were neglected. In addition, only low Mach numbers (up to 0.5) were considered.

42 Liu and Batill [38] developed a gradient-enhanced ANN method that does not re-  
43 quire weighting the residuals of the targets and gradients, but they only solved analytic  
44 benchmarks with up to 15 input variables. Pan and Duraisamy [44] used ANN to model  
45 nonlinear dynamical systems with gradient information. In that work, the gradient was  
46 penalized to obtain a smooth response instead of matching the gradient of the ANN  
47 model with the gradient of the data. Another approach using gradient-enhanced ANN  
48 was developed by Czarnecki et al. [15], where the loss function was defined as a weighted  
49 sum of the output estimation error and derivatives estimation error.

50 Airfoil shape design is one of the most fundamental elements in aerospace engineering.  
51 Given the current computer power, it is possible to compute steady flow for airfoils  
52 using CFD to solve RANS equation in minutes, which opens the door to performing the  
53 thousands of airfoil solutions required for machine-learning processes. Furthermore, since  
54 the adjoint method efficiently computes the gradient of performance metrics with respect  
55 to shape and flow variables efficiently, the computed gradients provide an additional rich

56 set of data. For a cost that is equivalent to less than a flow solution, the adjoint method  
57 provides a potentially large vector of data.

58 ANN frameworks that operate at large scale and in heterogeneous environments have  
59 improved dramatically in the last few years and are easier to use than ever. TensorFlow  
60 for example, has a high-level Python interface that has become popular owing to the  
61 combination of power and relative ease of installation and use [1].

62 Enabled by the above developments, we developed a new gradient-enhanced ANN  
63 approach and applied it to airfoil shape design optimization. We compared this newly-  
64 developed approach to the surrogate-model approach developed by Li et al. [35] for the  
65 prediction of the drag, lift, and moment coefficients. The surrogate-model approach  
66 required the division of the airfoil shape and flight condition into two flow regimes (sub-  
67 sonic and transonic) because the surrogate modeling techniques could not simultaneously  
68 achieve acceptable accuracy for both regimes. The ANN approach developed herein, how-  
69 ever, managed to model both subsonic and transonic airfoils in the whole flight regime  
70 with one single model.

71 The remainder of the manuscript is structured as follows. We describe the gradient-  
72 enhanced ANN and our improvements in Section 2. We present the results in Section 3,  
73 where we compare our previous surrogate model approach against the new gradient-  
74 enhanced ANN model. Finally, we present two airfoil shape optimization problems for  
75 the subsonic and transonic regimes in Section 4.

## 76 2. Gradient-enhanced ANN

77 In principle, using the training derivatives in addition to the training output values  
78 should improve the accuracy of the output prediction in surrogate modeling. Using this  
79 additional information should be even more advantageous when the computation of the  
80 gradients is efficient.

81 Many physics models now include adjoint methods. ADflow computes gradient infor-  
82 mation using the adjoint method, which computes the derivatives of a function of interest  
83 with respect to input variables at a cost that is independent of the number of those input  
84 variables [41, 33]. Similarly, DAFoam <sup>1</sup> which is developed based on openFoam, also has  
85 an adjoint implementation [27, 52]. This is in contrast to other gradient computation  
86 methods, such as finite-difference and complex-step methods, which scale linearly with  
87 the number of input variables [42].

88 In this work, we use ADflow, an open-source RANS CFD solver that includes an ad-  
89 joint method and efficient aerodynamic shape optimization [54, 33]. ADflow has been suc-  
90 cessfully used to perform aerodynamic shape optimization for airfoils [29, 35], wings [39,  
91 14, 55, 6, 40], and full aircraft configurations [14, 50, 49]. ADflow is also coupled to other  
92 disciplines to solve multidisciplinary design optimization problems [36, 11, 22, 18, 12, 28].

93 Our approach to exploiting the gradient information is based on the gradient-enhanced  
94 ANN method developed by Czarnecki et al. [15]. They introduced the Sobolev training  
95 for artificial neural networks (SANN) method that incorporates the gradient information  
96 in the loss function with the training samples during the training phase.

---

<sup>1</sup><https://github.com/mdolab/adflow>

97    2.1. Description of SANN

98    To approximate a function  $f$  using SANN, we use the matrix of training input values

$$99 \quad \mathbf{X} = [\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(n_t)}]^T, \quad (1)$$

100 where each training input is

$$101 \quad \mathbf{x}^{(i)} = [x_1^{(i)}, \dots, x_d^{(i)}]^T, \quad (2)$$

102 and hence  $\mathbf{X}$  is a  $(n_t \times d)$  matrix, where  $d$  is the number of dimensions of the input and  
103  $n_t$  is the number of training samples. The corresponding vector of training output values  
is

$$104 \quad \mathbf{y} = [y^{(1)}, \dots, y^{(n_t)}]^T, \quad (3)$$

105 where  $y^{(i)} = f(\mathbf{x}^{(i)})$ . Then, the gradient of the function that computes the outputs is

$$106 \quad \nabla f(\mathbf{x}^{(i)}) = \left[ \frac{\partial f}{\partial x_1}(\mathbf{x}^{(i)}), \dots, \frac{\partial f}{\partial x_d}(\mathbf{x}^{(i)}) \right]^T, \quad (4)$$

107 which we write as a matrix when evaluated for each training point as

$$108 \quad \mathbf{df} = \nabla f(\mathbf{X}) = [\nabla f(\mathbf{x}^{(1)}), \dots, \nabla f(\mathbf{x}^{(n_t)})]^T. \quad (5)$$

109 To train a neural network model  $m$  using a set of parameters  $\theta$ , we minimize a loss  
110 function with respect to  $\theta$ , i.e.,

$$111 \quad \begin{aligned} & \min_{\theta} y_{\text{loss}}(\mathbf{X}, \mathbf{y} | \theta), \\ & y_{\text{loss}}(\mathbf{X}, \mathbf{y} | \theta) := l(m(\mathbf{x}^{(i)} | \theta), y^{(i)}), \end{aligned} \quad (6)$$

112 where  $l$  is a loss function.

113 To incorporate the gradients into the training of an ANN, Czarnecki et al. [15] pro-  
114 posed adding a second term to the loss function above

$$115 \quad \begin{aligned} & \min_{\theta} \bar{y}_{\text{loss}}(\mathbf{X}, \mathbf{y}, \mathbf{df} | \theta), \\ & \bar{y}_{\text{loss}}(\mathbf{X}, \mathbf{y}, \mathbf{df} | \theta) := l(m(\mathbf{x}^{(i)} | \theta), y^{(i)}) + \sum_{j=1}^d l_j \left( \frac{\partial m}{\partial x_j}(\mathbf{x}^{(i)} | \theta), \frac{\partial f}{\partial x_j}(\mathbf{x}^{(i)}) \right), \end{aligned} \quad (7)$$

116 where  $l_j$  is the loss function with respect to the  $j^{\text{th}}$  partial derivative. In this equation,  
117 the second term involving the derivatives is dominant in the training of the SANN model,  
especially when  $d$  is high. This slows down the convergence of the SANN training because  
it emphasizes the minimization of the gradient error.

118    2.2. Efficient approach for training SANN

119 To accelerate the training convergence, we propose to introduce the gradient informa-  
120 tion gradually during the learning process. We do this by incorporating a parameter  $\lambda_k$ ,

118 with  $k \in \mathbb{N}^*$ , in Equation (7), yielding

$$\begin{aligned} & \min_{\theta} \bar{y}_{\text{loss}}(\mathbf{X}, \mathbf{y}, df|\theta), \\ & \bar{y}_{\text{loss}}(\mathbf{X}, \mathbf{y}, df|\theta) := \sum_{i=1}^{n_t} l(m(\mathbf{x}^{(i)}|\theta), y^{(i)}) + \\ & \quad \lambda_k \sum_{j=1}^d l_j \left( \frac{\partial m}{\partial x_j}(\mathbf{x}^{(i)}|\theta), \frac{\partial f}{\partial x_j}(\mathbf{x}^{(i)}) \right), \end{aligned} \quad (8)$$

119 where the parameter  $\lambda_k \in [0, 1]$  is a weight that controls how much gradient information  
120 to use in the SANN training.

121 We use TensorFlow [1], which is a large-scale machine learning framework, to compute  
122 the derivative  $\partial m / \partial x_j(\mathbf{x}^{(i)}|\theta)$ . TensorFlow uses reverse mode automatic differentiation  
123 to compute the derivatives.

124 We call this approach modified SANN (mSANN). The key idea is to gradually increase  
125  $\lambda$  for each epoch in the training. Algorithm 1 describes the main steps in mSANN.

---

**Algorithm 1:** mSANN

---

```

Input: ( $\mathbf{X}, \mathbf{y}, df, N$ ); //  $N$  is the total number of epochs
Define the ANN architecture; // Layers, units, and activation functions
Define  $\lambda_k$ ; // List containing values of  $\lambda$ 
Initialize  $\theta$  in Equation (8); // Use a normal random variable  $\mathcal{N}(0, 0.1)$ 
for  $i \leq N$  do
    for  $\lambda \in list(\lambda_k)$  do
        | Solve minimization problem (8); // Use backpropagation algorithm
        | end
    end
Output:  $\hat{y}(\mathbf{x})$ ; // Final prediction

```

---

126 For a better illustration of the benefits of this approach, we use the Rosenbrock  
127 function

$$\sum_{i=1}^{15} [(x_{i+1} - x_i^2)^2 + (x_i - 1)^2], \quad -2 \leq x_i \leq 2, \quad \text{for } i = 1, \dots, 16. \quad (9)$$

128 to compare the performance of our approach with SANN and a standard ANN method  
129 without gradients.

130 We first build similar ANN architectures for all methods using eight layers: one input  
131 layer, six hidden layers, and one output layer. Figure 1, which is generated using the  
132 NN-SVG online tool <sup>2</sup>, depicts the architecture and activation functions of the ANN. In  
133 general, there is no rule of thumb to know in advance the best ANN architecture . In  
134 our case, the architectures are chosen based on trial and error after many experiments.  
135 Similar to the ANN architecture, the tuned parameters for training an ANN model, such

---

<sup>2</sup><http://alexlenail.me/NN-SVG/LeNet.html>

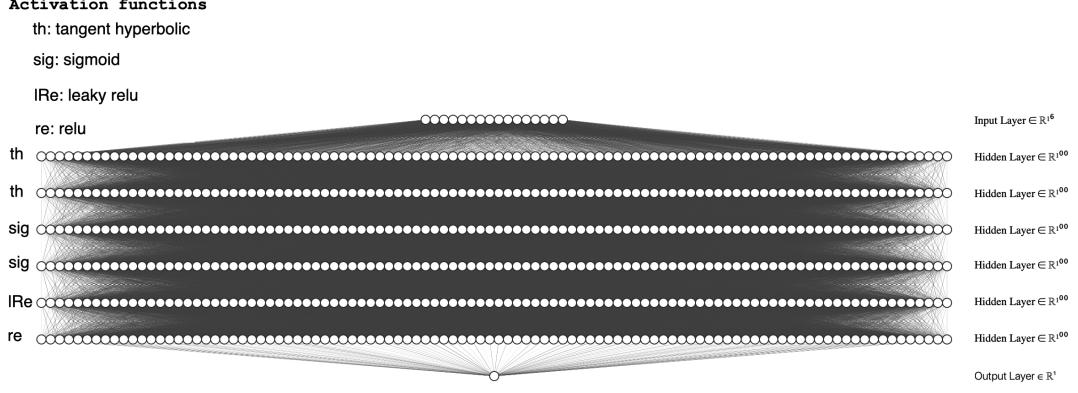


Figure 1: This ANN architecture is composed of one input layer, six hidden layers, and one output layer. The activation functions are indicated next to the left edge of each hidden layer. The number of units for each hidden layer is 100.

as the learning rate, the number of layers, the number of units, and so on, is problem-dependent. Tuning the parameters  $\lambda_k$  for solving Equation (8) is also problem-dependent. In our case, we set  $\lambda_k = 0, 0.1, 0.2, \dots, 1$ . We use TensorFlow to construct the ANN, SANN, and mSANN models.

After building the ANN, we sample 42,000 training points and 22,000 validation points using the Latin hypercube sampling method implemented by Bouhlel et al. [10]. Then, we use the  $L_2$ -norm ( $\|\cdot\|_2$ ) for both  $l$  and  $l_j$ . Finally, we use three criteria to compare between the various ANNs:

- The  $y_{\text{loss}}$  and  $\bar{y}_{\text{loss}}$  for each epoch at the training samples, which shows the evolution of the training process. The evolution of  $\bar{y}_{\text{loss}}$  is only applicable for SANN and mSANN.
- The  $L_2$ -norm of relative errors ( $\epsilon_{L_2}$ ) for both training and validation samples

$$\epsilon_{L_2} = \frac{\|\hat{\mathbf{y}} - \mathbf{y}\|_2}{\|\mathbf{y}\|_2} 100, \quad (10)$$

where  $\mathbf{y}$  is a vector containing the real values and  $\hat{\mathbf{y}}$  is a vector containing the prediction of  $\mathbf{y}$ . This provides a measure of the final solution accuracy.

- The  $L_2$ -norm of relative errors ( $\bar{\epsilon}_{L_2}$ ) of the gradient predictions for both training and validation samples

$$\bar{\epsilon}_{L_2} = \frac{\|\mathbf{d}\hat{\mathbf{y}} - \mathbf{dy}\|_2}{\|\mathbf{dy}\|_2} 100, \quad (11)$$

where  $\mathbf{dy}$  is a vector containing the real values of the gradient and  $\hat{\mathbf{dy}}$  is a vector containing the prediction of  $\mathbf{dy}$ . The vectors  $\mathbf{dy}$  and  $\hat{\mathbf{dy}}$  are shaped in a way to contain all the derivative values into one vector column; i.e.,  $\mathbf{dy}$  and  $\hat{\mathbf{dy}}$  are both vectors for training and validation, respectively.

Figure 2 compares the results of ANN, SANN and mSANN models. The  $y_{\text{loss}}$  and  $\bar{y}_{\text{loss}}$  loss functions drop off rapidly with mSANN especially for the first epochs (an epoch in

Table 1: The  $L_2$ -norm of the relative errors results given by Equations (10) and (11) for both training and validation on the Rosenbrock function.

	Training		Testing	
	$\epsilon_{L_2}$	$\bar{\epsilon}_{L_2}$	$\epsilon_{L_2}$	$\bar{\epsilon}_{L_2}$
mSANN	1.20	5.27	1.25	5.33
SANN	4.00	14.18	4.06	14.20
ANN	4.40	25.50	4.78	25.60

158 ANN is one forward pass and one backward pass of all the training examples). In addition,  
 159 mSANN shows a better convergence in all cases by at least two orders of magnitude. The  
 160 final solutions obtained are: 8.88 (mSANN), 11.30 (SANN), and 11.52 (ANN) for  $y_{loss}$   
 161 and 11.57 (mSANN) compared to 13.60 (SANN) for  $\bar{y}_{loss}$ .

162 As shown in Table 1, mSANN outperforms ANN and SANN in both the prediction  
 163 of the output of interest and the prediction of the gradient for training and validation  
 164 samples.

### 165 3. Application to airfoil analysis

166 As previously mentioned, Li et al. [35] developed a data-based approach to provide  
 167 fast aerodynamic analysis and design optimization tool for airfoils in the subsonic and  
 168 transonic flow regimes using two separate surrogate models for each of these flow regimes.  
 169 In this section, we first describe the methodology followed by Li et al. [35]. Then, we  
 170 describe our approach for constructing a fast airfoil analysis and design tool in both  
 171 subsonic and transonic using a single mSANN model. We compare the performance of the  
 172 mSANN model with the results given of Li et al. [35] for the subsonic case and demonstrate  
 173 the performance of mSANN in the transonic regime. We compare the mSANN to the  
 174 GE-KPLS method only on particular airfoil cases in transonic regime. This is because Li  
 175 et al. [35] developed a surrogate model in a small design space in transonic regime while  
 176 our developed method covers a much larger design space.

#### 177 3.1. Data-based approach in subsonic flow regime

178 The data-based approach developed by Li et al. [35] used a new airfoil shape param-  
 179 eterization defined by camber-thickness mode shapes. They first selected 1172 subsonic  
 180 airfoils from the Webfoil database <sup>3</sup> to cover as large an airfoil design space as possible.  
 181 Then, they computed the camber-thickness mode shapes of these airfoils using the singu-  
 182 lar value decomposition method. Seven camber modes and seven thickness modes were  
 183 sufficient to accurately regenerate the initial geometry of the airfoils. This reduces the  
 184 number of the airfoil shape parameters to 14 while covering a large design space with dif-  
 185 ferent geometry complexities of airfoils. The free-stream Mach number ( $M$ ) was allowed  
 186 to vary between 0.3 and 0.6 while the angle of attack ( $\alpha$ ) ranged from  $-2$  to  $6$  degrees.

187 They generated new sampling airfoils data into many clusters (defined by dividing the  
 188 design space into sub-domains with respect to  $M$ ,  $\alpha$ , one aerodynamic coefficient, and  
 189 the first camber and thickness modes) with well-defined bounds of mode shapes.

---

<sup>3</sup><http://webfoil.engin.umich.edu>

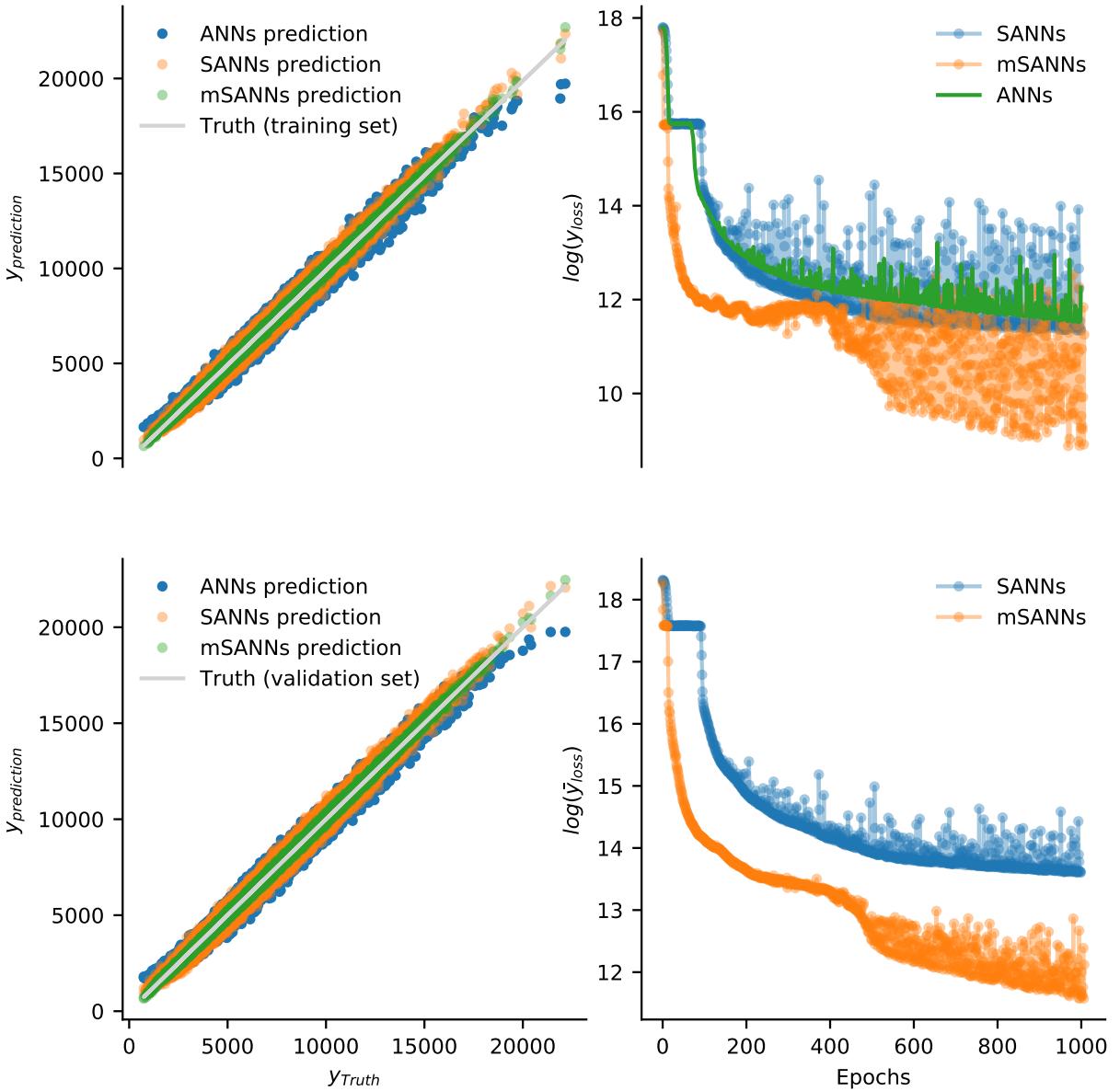


Figure 2: Rosenbrock function results. Upper left: comparison between the training predictions of ANN, SANN, and mSANN. The best predictions are close to the diagonal (black line). mSANN shows a better prediction on the training samples than both ANN and SANN. Bottom left: Comparison between the validation predictions. Similar to the training predictions, we obtain better predictions using mSANN. Upper right: The mSANN model reduces the loss function faster than ANN and SANN especially during the first iterations, and yields to a better convergence. Bottom right: mSANN is faster and exhibits better convergence than SANN.

190 The airfoil aerodynamic force coefficients and their respective gradients are computed  
 191 using ADflow, which solves the RANS equations with a Spalart–Allmaras turbulence  
 192 model [54].<sup>4</sup> The RANS CFD meshes for all airfoils are automatically generated with the  
 193 open-source package pyHyp.<sup>5</sup> The flow solution residuals are converged by 15 orders of  
 194 magnitude below the initial residual using an approximate Newton–Krylov method at  
 195 the initial stage and then switching to an exact Newton–Krylov method at the final stage

196 Since the gradient information is required for the construction of the gradient-en-  
 197 hanced surrogate model, the ADflow adjoint solver [33] is also called to compute the  
 198 gradient with respect to all modes, Mach number, and angle of attack. Later we also use  
 199 this data set to train the Neural Network presented in Section 3.2.

200 Finally, Li et al. [35] used a mixture of experts method [25, 4] with three main steps:

- 201 1. Train a local gradient-enhanced surrogate model for each cluster, using gradient-  
 202 enhanced kriging combined with partial-least-squares model (GE-KPLS) [7]. This  
 203 surrogate model scales well in inputs and it is among the few gradient-enhanced  
 204 kriging-based surrogates that scale well with training data.
- 205 2. Compute the cluster posterior probability for each cluster with a supervised learning  
 206 algorithm to compute the proportion of each cluster using a regularized Gaussian  
 207 classifier Liem et al. [37].
- 208 3. Build a surrogate model as a linear combination of all local surrogate models using  
 209 the mixing proportion computed in Step 2.

210 For simplicity, this methodology is named GE-KPLS in this section because it is based  
 211 on the GE-KPLS model.

### 212 3.2. *mSANN approach for both subsonic and transonic regimes*

213 Li et al. [35] used two different shape parametrizations for the subsonic and transonic  
 214 airfoils. Instead of the seven camber and seven thickness mode shapes used for the  
 215 subsonic airfoils, the shape parameterization for the transonic airfoils used four camber  
 216 modes and four thickness modes, which have no relation to the subsonic ones.

217 Since we want to use both databases to create a single model using the mSANN  
 218 approach, we need to find a way to deal with the distinct sets of mode shapes. The  
 219 common inputs in subsonic and transonic databases are  $M$  and  $\alpha$ . To fuse both pa-  
 220 rameterizations, we developed a sparse geometry parametrization. For each regime, we  
 221 have matrices of data ( $\mathbf{X}$ ) with size  $(n \times d)$ , where  $n$  is the number of samples and  $d$  is  
 222 the number of input variables (two flow conditions plus the camber and thickness mode  
 223 coefficients). We decompose each matrix as  $\mathbf{X} = [\mathbf{X}_{ct}, \mathbf{X}_{\alpha M}]$  to separate the values for  
 224 the mode coefficients ( $ct$ ) and flow conditions ( $\alpha M$ ). Then, we merge the matrices corre-  
 225 sponding to the subsonic (sub) and transonic (trans) data into one matrix with a size of  
 226  $(n_{sub} + n_{trans}) \times (d_{sub} + d_{trans} - 2)$  containing both subsonic and transonic data to yield

$$\mathbf{X} = \begin{bmatrix} \mathbf{X}_{ct_{trans}} & \mathbf{0} & \mathbf{X}_{\alpha M_{trans}} \\ \mathbf{0} & \mathbf{X}_{ct_{sub}} & \mathbf{X}_{\alpha M_{sub}} \end{bmatrix}, \quad (12)$$

---

<sup>4</sup><https://github.com/mdolab/adflow>

<sup>5</sup><https://github.com/mdolab/pyhyp>

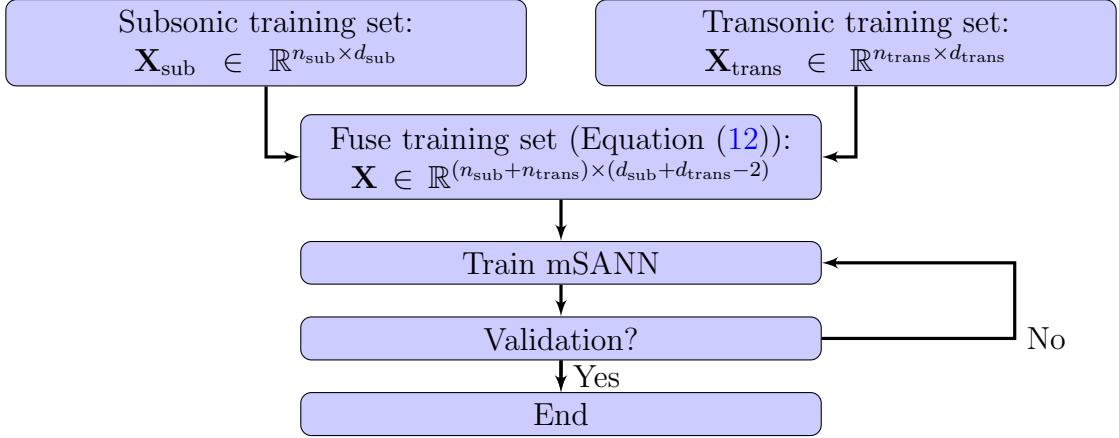


Figure 3: Workflow of training mSANN for both subsonic and transonic regimes.

where the two matrices of zeros have the appropriate sizes to fill the corresponding blocks. The flowchart in Figure 3 shows the workflow of training mSANN for both the subsonic and transonic regimes. First, we set both the subsonic and transonic training sets. Second, we fuse both training sets into a single training set  $\mathbf{X}$  using Equation (12). Third, we run the training of mSANN on  $\mathbf{X}$ . Finally, we check the validation of the model: if the desired accuracy is reached we stop the training, otherwise, we continue the training of the model.

Our objective is to train an mSANN model using both subsonic and transonic databases to predict the aerodynamic force and moment coefficients:  $C_l$ ,  $C_d$ , and  $C_m$ .

### 3.3. Validation procedure

We use a mSANN architecture similar to the one shown in Figure 1 with  $\lambda_k = 0, 0.1, 0.2, \dots, 0.9, 1$ . There are 42,039 and 4,120 training samples for the subsonic and transonic regimes, respectively. The training database used for mSANN in the subsonic regime is a subset of the one used for GE-KPLS by Li et al. [35], but the testing and validation samples are the same. To validate the performance of the mSANN model, we use 21,870 and 1,544 validation samples for subsonic and transonic regimes, respectively. To verify the generalization of the model, the validation set is split into two smaller sets: 10,935 validation samples and the same number of test samples for the subsonic regime, and 787 validation samples and 757 test samples for the transonic regime. If the accuracy of the test set is sufficient, we stop the training; otherwise, we continue the training. For the subsonic set,  $M \in [0.3, 0.6]$  and  $\alpha \in [-0.5, 6]$  degrees, while for the transonic set  $M \in [0.7, 0.75]$  and  $\alpha \in [-1.5, 4.5]$  degrees.

To assess the performance of the mSANN model, we compute the  $L_2$ -norm of the relative error (10) for both validation and test sets. We also compute the  $L_1$ -norm of the relative error of each sample (for both validation and test samples), which is given by

$$\epsilon_{L_1}^i = \left| \frac{y_{\text{pred}}^{(i)} - y_{\text{true}}^{(i)}}{y_{\text{true}}^{(i)}} \right|. \quad (13)$$

In the subsonic regime, we compare the global performance of mSANN to GE-KPLS on the same testing and validation sample sets. In addition, we show the performance of only mSANN in transonic regime. In addition, we compare the aerodynamic

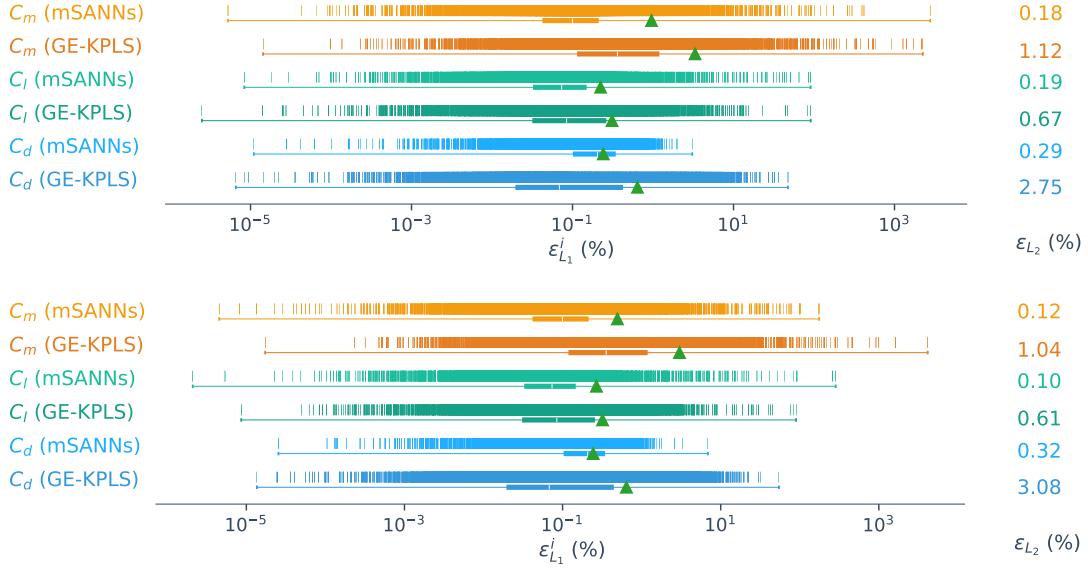


Figure 4: Comparison of the  $L_1$ -norm of the relative errors and the  $L_2$ -norm of the relative errors between GE-KPLS and mSANN for validation (top) and testing (bottom) in subsonic flow regime. The mean of the  $L_1$ -norm of the relative errors are given by the triangles. The white line in the box represents the median of the  $L_1$ -norm of the relative errors. mSANN yields to better results than the GE-KPLS model, especially for the drag coefficient.

coefficients curves versus angle of attack between the GE-KPLS and mSANN models on three selected subsonic airfoils with different geometry complexities. For transonic airfoil optimization, Li et al. [35] optimized three transonic airfoils with  $M = 0.73$ . We compare the performance of mSANN to GE-KPLS on the same transonic airfoils used by Li et al. [35].

### 3.4. Comparison between GE-KPLS and mSANN in subsonic flow regime

GE-KPLS and mSANN are compared in Figure 4. The mSANN model yields to better results than GE-KPLS for all aerodynamic coefficients for both validation and testing, especially for  $C_d$  (e.g.; 0.29% error for mSANN versus 2.75% for GE-KPLS in terms of  $\epsilon_{L_2}$ ). For  $C_d$  in the test samples, there is an improvement in maximum  $\epsilon_{L_1}^i$  of about 1.5 orders of magnitude. Moreover, mSANN yields a better mean of the  $L_1$ -norm of the relative errors in all cases, especially for  $C_m$ , where there is an improvement of almost one order of magnitude. This shows that gradient-enhanced neural networks are able to learn complex models with different physical behavior over the design space.

We select three airfoils of different types that do not exist in the database used for training the model: NACA 4412 (max thickness 12% at 30% chord and max camber 4% at 40% chord), Clark Y (max thickness 11.7% at 28% chord and max camber 3.4% at 42% chord), and NLF(1)-1015 (max thickness 15% at 39.8% chord and max camber 4.3% at 62.8% chord). Figure 5 shows the airfoils shapes and their aerodynamic coefficient curves. For  $C_l$  and  $C_d$ , the mSANN and GE-KPLS curves match all the CFD simulations. For  $C_m$ , mSANN outperforms GE-KPLS on both NACA 4412 and NLF(1)-1015, especially for the last airfoil, where the GE-KPLS over-predicts compared to the CFD simulations. For example, we have an error of 0.01% with mSANN versus an error of 1.91% with

278 GE-KPLS for NLF(1)-1015 at  $\alpha = 5.87$ . In addition, the mSANN curve is smoother  
279 compared to GE-KPLS in both cases. However, mSANN is slightly worse than GE-  
280 KPLS in predicting  $C_m$  near  $\alpha = 6^\circ$  for the first two airfoils. Indeed, for NACA 4412 for  
281 example, we observe an error of 0.4% and 0.6% with mSANN versus an error of 0.07%  
282 and 0.1% with GE-KPLS at  $\alpha = 5.55$  and 5.91, respectively.

### 283 *3.5. Performance of mSANN in transonic flow regime*

284 The distribution of the  $\epsilon_{L_1}^i$  and  $\epsilon_{L_2}$  values are given in Figure 6. The  $L_2$ -norm of  
285 the relative errors of the model increases slightly compared to the subsonic results given  
286 in Figure 4. However, the accuracy of the mSANN model is within the same order of  
287 magnitude for both subsonic and transonic regimes.

288 In addition, the results of the test and validation in terms of  $\epsilon_{L_2}$  are approximately  
289 the same: the  $L_2$ -norm of the relative errors on the test samples is slightly larger than  
290 the error on the validation samples (with a difference of 0.07% for both  $C_d$  and  $C_m$ ). On  
291 the other hand, the means of the  $L_1$ -norm of the relative errors are very similar in this  
292 case. In sum, our approach generalizes well in both subsonic and transonic regimes.

293 Figure 7 compares GE-KPLS and mSANN for three transonic NASA airfoils: SC(2)-  
294 0710 (max thickness 10% at 37% chord and max camber 2.1% at 81% chord), SC(2)-0606  
295 (max thickness 6% at 34% chord and max camber 1.3% at 81% chord), and SC(2)-0404  
296 (max thickness 4% at 37% chord and max camber 0.7% at 81% chord). The mSANN  
297 model outperforms GE-KPLS in all cases, especially in predicting  $C_m$ , where GE-KPLS  
298 mismatches most of the CFD simulations. In addition, mSANN provides smoother curves,  
299 especially when we compare mSANN to GE-KPLS for  $C_m$  with SC(2)-0710.

300 These results show the ability of our approach to approximate two different design  
301 spaces (subsonic and transonic) simultaneously.

## 302 **4. Airfoil shape optimization**

303 Using the mSANN model trained as described in Section 3.2, we perform two airfoil  
304 shape optimizations in the subsonic and transonic flow regimes. We couple the mSANN  
305 model to SNOPT (Sparse Nonlinear OPTimizer), a gradient-based optimizer that imple-  
306 ments sequential quadratic programming and solves large-scale constrained optimization  
307 problems efficiently [20]. To verify the performance of our optimization results, we com-  
308 pare the solutions provided by the mSANN-based optimization to optimizations using  
309 direct the CFD evaluations.

### 310 *4.1. Subsonic airfoil optimization*

311 In the subsonic case, we minimize the drag coefficient with respect to 14 modes and  
312 angle of attack (fixing  $M = 0.45$ ) subject to a constraint  $C_l = 0.5$ , starting from the  
313 NACA 0012 airfoil shape. To avoid abnormal airfoils, we constraint the upper bound  
314 of the first thickness mode to be no more than 90% of the NACA 0012 first thickness  
315 mode. The remaining 13 mode shape bounds are computed as described by Li et al. [35].  
316 Therefore, the total number of constraints is 15 (14 constraints on the mode coefficients  
317 and 1 constraint on  $C_l$ ).

318 Figure 8 compares the solution found by mSANN with the CFD reference solution.  
319 The grey area shows the design space range. In this case, the optimized airfoils with

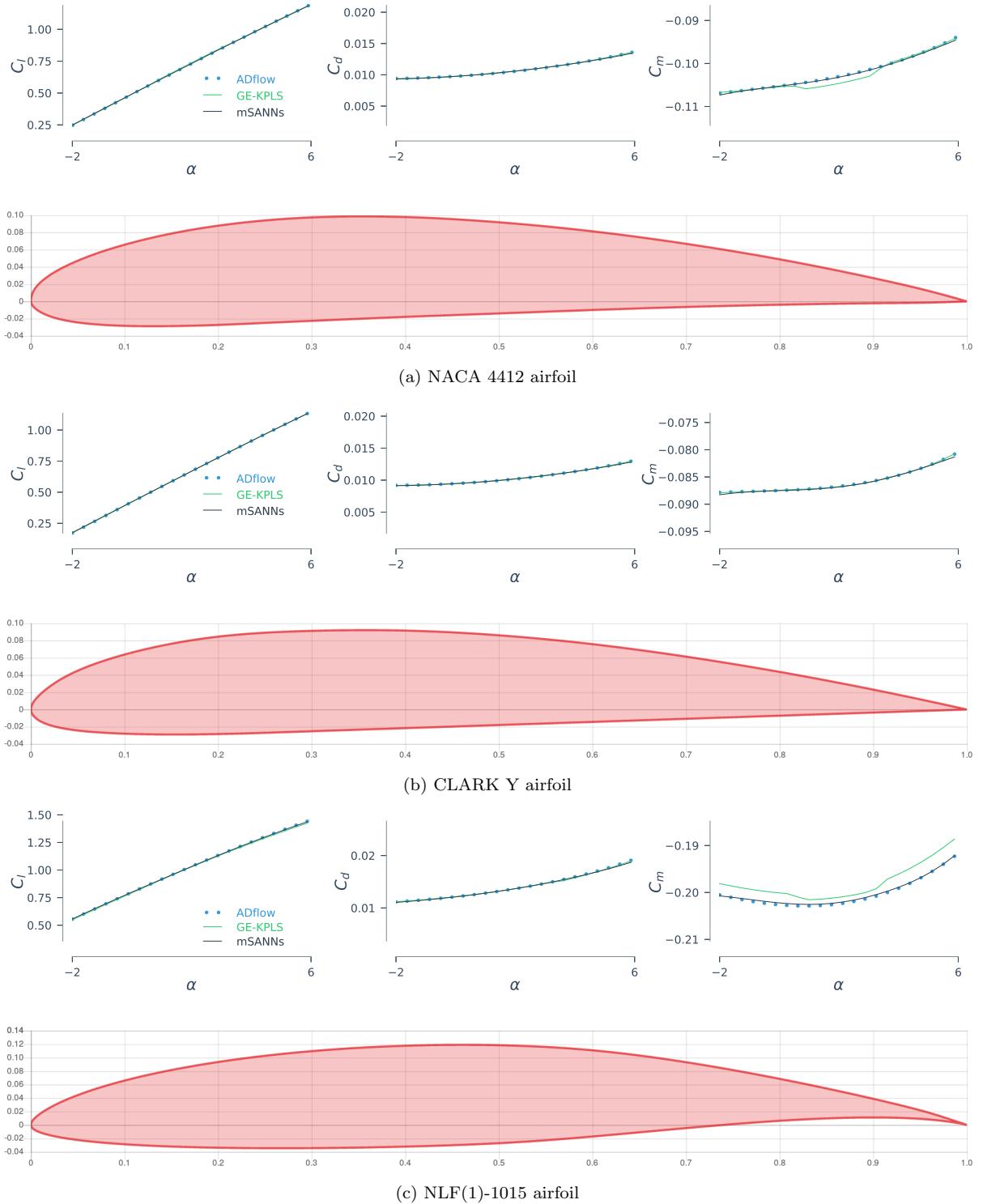


Figure 5: Comparison of aerodynamic coefficient curves between mSANN and GE-KPLS in subsonic ( $M = 0.45$ ) for three airfoils. The results of mSANN and GE-KPLS models are similar and match the CFD simulations, except for  $C_m$ , where mSANN performs better overall.

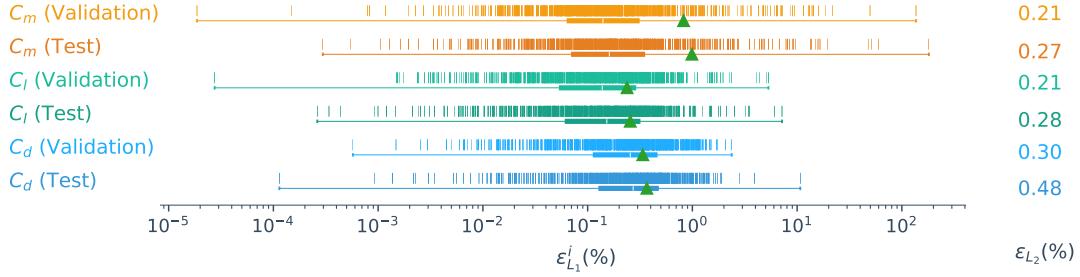


Figure 6: Results of the  $L_1$ -norm of the relative errors and the  $L_2$ -norm of the relative errors for validation and test samples in transonic. The mean of the  $L_1$ -norm of the relative errors are shown by the triangles. The white line in the box represents the median of the  $L_1$ -norm of the relative errors.

320 mSANN and CFD are almost identical. When comparing the CFD-computed coefficients,  
321  $C_d$  differs by only 0.01 counts and  $C_l$  by 0.001.

#### 322 4.2. Transonic airfoil optimization

323 In the transonic case, we minimize the  $C_d$  coefficient with respect to eight mode shapes  
324 and angle of attack at  $M = 0.72$  subject to a lift constraint set to  $C_l = 0.82$ . Similar  
325 to the subsonic case, we constrain the upper bound of the first thickness mode to be no  
326 more than 90% of the SC(2)-0710 first thickness mode. The remaining seven mode shape  
327 bounds are computed in a similar way to the subsonic case. Therefore, the total number  
328 of constraints is 9 (8 constraints on the mode coefficients and 1 constraint on  $C_l$ ).

329 Figure 9 compares the two optimal airfoils. Transonic airfoil shape optimization is  
330 more challenging than subsonic airfoil shape optimization due to the much more nonlinear  
331 flow physics. Nevertheless, the mSANN-based optimization achieves an airfoil similar to  
332 the reference one. The  $C_d$  differs by only 0.12 counts and the difference in  $C_l$  is only  
333 0.002.

#### 334 4.3. Efficiency of the mSANN-based optimization algorithm

335 The cost of an ADflow RANS simulation is between 2 and 5 minutes. Computing  
336 a large number of those simulations is expensive, so the training set used for training  
337 mSANN is between 58 and 145 days for the subsonic regime and between 5 and 14 days  
338 for the transonic one. We reduce this cost using parallel evaluations, resulting in a train-  
339 ing cost for mSANN between 8 to 10 hours. The cost of a mSANN-based optimization is  
340 only a few seconds.

341 An optimization based on direct calls to ADflow costs between 20 and 60 minutes.  
342 This means that the total cost of a few mSANN-based optimizations is much more ex-  
343 pensive than the ADflow-based ones when we take into account the training cost, this  
344 large cost can be amortized as the number of optimizations increases beyond 100 or so.

345 Finally, the mSANN-based optimization approach made it possible to create the opti-  
346 mization capability in Webfoil<sup>6</sup>, which provides an interactive tool for airfoil design that  
347 would not be possible with direct calls to ADflow.

---

<sup>6</sup><http://webfoil.engin.umich.edu>

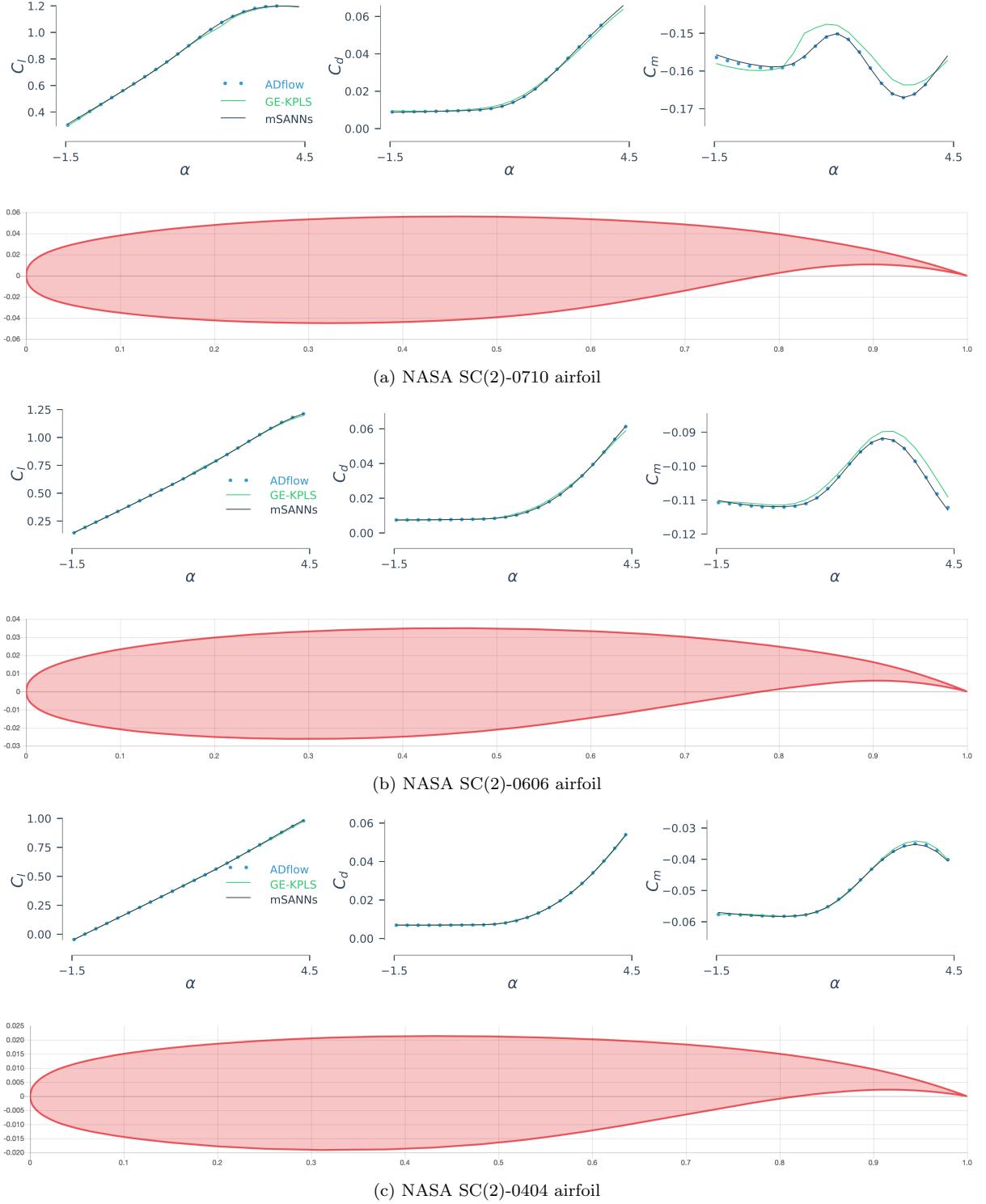


Figure 7: Comparison between mSANN and GE-KPLS models in the transonic regime ( $M = 0.73$ ) for three airfoils.

	$C_d$ (count)		$C_l$		$\alpha$
	ADflow	mSANN	ADflow	mSANN	
NACA0012	97.21	97.17	0.500	0.491	3.986
$Opt_{ADflow}$	<b>92.17</b>	<b>92.36</b>	<b>0.499</b>	<b>0.499</b>	<b>1.863</b>
$Opt_{mSANN}$	<b>92.18</b>	<b>92.36</b>	<b>0.500</b>	<b>0.500</b>	<b>1.864</b>

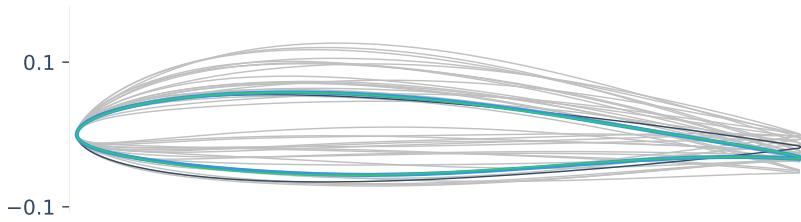


Figure 8:  $C_d$  minimization at  $C_l = 0.5$  and  $M = 0.45$ . The optimized solutions given by ADflow and mSANN are in blue and green, respectively. The grey area shows the design space range. The optimal airfoil found using mSANN is indistinguishable to the one found using CFD.

	$C_d$ (count)		$C_l$		$\alpha$
	ADflow	mSANN	ADflow	mSANN	
SC(2) – 0710	114.44	1143.87	0.819	0.817	1.343
$Opt_{ADflow}$	<b>101.31</b>	<b>101.36</b>	<b>0.820</b>	<b>0.818</b>	<b>0.848</b>
$Opt_{mSANN}$	<b>101.43</b>	<b>101.47</b>	<b>0.822</b>	<b>0.820</b>	<b>0.864</b>

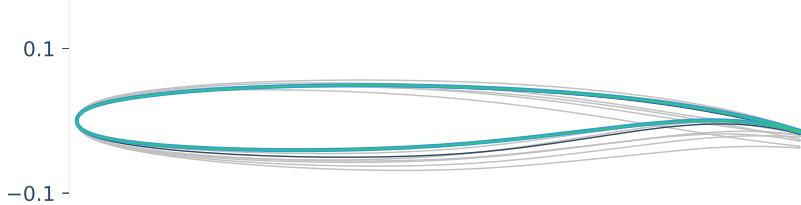


Figure 9:  $C_d$  minimization at  $C_l = 0.82$  and  $M = 0.72$ . The optimized solutions given by ADflow and mSANN are in blue and green, respectively. The gray area shows the design space range. The optimal airfoil found using mSANN is indistinguishable to the one found using CFD.

348 **5. Replication results**

349 The codes needed for reproducing the results in this paper are available online un-  
350 der open-source licenses. After installing TensorFlow, the scripts to generate the re-  
351 sults presented in this paper are available on [https://data.mendeley.com/datasets/  
ngpd634smf/1](https://data.mendeley.com/datasets/ngpd634smf/1). The Rosenbrock function test results given in Section 2.2 could be repro-  
352 duced using files available in the repository "Rosenbrock". The scripts constructing the  
353 mSANN and predicting the aerodynamic coefficients given in Section 3.2 are available in  
354 the repository "Analysis". The optimization results given in Section 4 could be repro-  
355 duced using the script files hosted in the repository "Optimization". In addition, ADflow  
356 is available on <https://github.com/mdlolab/adflow> for reproducing ADflow results.  
357

358 **6. Conclusions**

359 We developed an ANN methodology (mSANN) for fast and accurate airfoil design  
360 applications. The ANN model was enhanced with gradient information to increase the  
361 model accuracy for a given number of training samples. mSANN is based on gradually  
362 introducing the gradient information during the learning process via a dynamic parameter  
363  $\lambda$ . We validated mSANN on an academic test case, the Rosenbrock function, and showed  
364 that it outperforms the standard ANN model and SANN model with an improvement  
365 of about 3% in terms of the  $L_2$ -norm of relative error. This approach is applicable in a  
366 large design space containing over one thousand airfoil shapes of various types at flow  
367 conditions ranging from subsonic to the transonic regime.

368 In previous work, we had developed a gradient-enhanced Kriging surrogate model  
369 (GE-KPLS) for the same purpose. However, that approach required separate models for  
370 the subsonic and transonic regimes. With mSANN, we were able to create a single-mode  
371 covering both regimes that is even more accurate. In subsonic, we obtained an error of  
372 0.29% in terms of  $L_2$ -norm of relative errors compared to an error of 2.75% using GE-  
373 KPLS for  $C_d$ . The  $L_2$ -norm of relative errors for  $C_d$  in the transonic regime is 0.48%  
374 compared to the CFD results.

375 We performed two airfoil shape optimizations: one for the subsonic flow regime and  
376 another for the transonic regime. Compared to a CFD-based optimization, mSANN  
377 found the reference solutions with a difference of 0.01 counts and 0.12 counts for  $C_d$  in  
378 subsonic and transonic regimes, respectively.

379 These optimization problems are solved in less than four seconds instead of the hour  
380 required using the conventional approach that calls CFD directly, which opens the door to  
381 interactive airfoil design. This approach is now accessible through Webfoil, a web-based  
382 airfoil design tool and database.<sup>7</sup>

383 **Acknowledgments**

384 This work was partially supported by the Air Force Office of Scientific Research  
385 (AFOSR) MURI on "Managing multiple information sources of multi-physics systems",  
386 Program Officer Jean--Luc Cambier, Award Number FA9550-15-1-0038.

---

<sup>7</sup><http://webfoil.engin.umich.edu>

387 **References**

- 388 [1] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat,  
389 G. Irving, M. Isard, M. Kudlur, J. Levenberg, R. Monga, S. Moore, D. G. Murray,  
390 B. Steiner, P. Tucker, V. Vasudevan, P. Warden, M. Wicke, Y. Yu, and X. Zheng.  
391 Tensorflow: A system for large-scale machine learning. In *12th USENIX Symposium  
392 on Operating Systems Design and Implementation (OSDI 16)*, pages 265–  
393 283, 2016. URL [https://www.usenix.org/system/files/conference/osdi16/  
394 osdi16-abadi.pdf](https://www.usenix.org/system/files/conference/osdi16/osdi16-abadi.pdf).
- 395 [2] A. Angelova, A. Krizhevsky, and V. Vanhoucke. Pedestrian detection with a large-  
396 field-of-view deep network. In *Proceedings of ICRA 2015*, 2015.
- 397 [3] J. Ba, V. Mnih, and K. Kavukcuoglu. Multiple object recognition with visual attention,  
398 2014. URL <http://arxiv.org/abs/1412.7755>. cite arxiv:1412.7755.
- 399 [4] D. Bettebghor, N. Bartoli, S. Grihon, J. Morlier, and M. Samuelides. Surrogate  
400 modeling approximation using a mixture of experts based on em joint estimation.  
401 *Structural and Multidisciplinary Optimization*, 43(2):243–259, 2011. doi:  
402 10.1007/s00158-010-0554-2.
- 403 [5] S. Bhatrai, J. H. de Baar, and A. J. Neely. Efficient uncertainty quantification for  
404 a hypersonic trailing-edge flap, using gradient-enhanced kriging. *Aerospace Science  
405 and Technology*, 80:261 – 268, 2018. ISSN 1270-9638. doi: 10.1016/j.ast.2018.06.036.
- 406 [6] N. P. Bons, X. He, C. A. Mader, and J. R. R. A. Martins. Multimodality in aerody-  
407 namic wing design optimization. *AIAA Journal*, 57(3):1004–1018, March 2019. doi:  
408 10.2514/1.J057294.
- 409 [7] M. A. Bouhlel and J. R. R. A. Martins. Gradient-enhanced kriging for high-  
410 dimensional problems. *Engineering with Computers*, 1(35):157–173, January 2019.  
411 doi: 10.1007/s00366-018-0590-x.
- 412 [8] M. A. Bouhlel, N. Bartoli, J. Morlier, and A. Otsmane. An improved approach for  
413 estimating the hyperparameters of the kriging model for high-dimensional problems  
414 through the partial least squares method. *Mathematical Problems in Engineering*,  
415 2016. doi: 10.1155/2016/6723410. Article ID 6723410.
- 416 [9] M. A. Bouhlel, N. Bartoli, A. Otsmane, and J. Morlier. Improving kriging sur-  
417rogates of high-dimensional design models by partial least squares dimension re-  
418 duction. *Structural and Multidisciplinary Optimization*, 53(5):935–952, 2016. doi:  
419 10.1007/s00158-015-1395-9.
- 420 [10] M. A. Bouhlel, J. T. Hwang, N. Bartoli, R. Lafage, J. Morlier, and J. R. R. A.  
421 Martins. A Python surrogate modeling framework with derivatives. *Advances in  
422 Engineering Software*, 135:102662, September 2019. doi: 10.1016/j.advengsoft.2019.  
423 03.005.
- 424 [11] T. R. Brooks, G. K. W. Kenway, and J. R. R. A. Martins. Benchmark aerostructural  
425 models for the study of transonic aircraft wings. *AIAA Journal*, 56(7):2840–2855,  
426 July 2018. doi: 10.2514/1.J056603.

- [427] [12] T. R. Brooks, J. R. R. A. Martins, and G. J. Kennedy. High-fidelity aerostructural  
 optimization of tow-steered composite wings. *Journal of Fluids and Structures*, 88:  
 122–147, July 2019. doi: 10.1016/j.jfluidstructs.2019.04.005.
- [430] [13] M. D. Buhmann and M. D. Buhmann. *Radial Basis Functions*. Cambridge University  
 Press, New York, NY, USA, 2003. ISBN 0521633389.
- [432] [14] S. Chen, Z. Lyu, G. K. W. Kenway, and J. R. R. A. Martins. Aerodynamic shape  
 optimization of the Common Research Model wing-body-tail configuration. *Journal  
 of Aircraft*, 53(1):276–293, January 2016. doi: 10.2514/1.C033328.
- [435] [15] W. M. Czarnecki, S. Osindero, M. Jaderberg, G. Swirszcz, and R. Pascanu. Sobolev  
 training for neural networks. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, pages 4281–4290, 2017. URL <http://papers.nips.cc/paper/7015-sobolev-training-for-neural-networks>.
- [440] [16] J. H. S. de Baar, T. P. Scholcz, and R. P. Dwight. Exploiting adjoint derivatives  
 in high-dimensional metamodels. *AIAA Journal*, 53(5):1391–1395, May 2015. doi:  
 10.2514/1.j053678.
- [443] [17] A. Dumont, J.-L. Hantrais-Gervois, P.-Y. Passaggia, J. Peter, I. Salah el Din,  
 and É. Savin. *Ordinary Kriging Surrogates in Aerodynamics*, pages 229–245.  
 Springer International Publishing, 2019. ISBN 978-3-319-77767-2. doi: 10.1007/  
 978-3-319-77767-2\_14.
- [447] [18] N. Garg, B. W. Pearce, P. A. Brandner, A. W. Phillips, J. R. R. A. Martins, and  
 Y. L. Young. Experimental investigation of a hydrofoil designed via hydrostructural  
 optimization. *Journal of Fluids and Structures*, 84:243–262, January 2019. doi:  
 10.1016/j.jfluidstructs.2018.10.010.
- [451] [19] K. Giannakoglou, D. Papadimitriou, and I. Kampolis. Aerodynamic shape design  
 using evolutionary algorithms and new gradient-assisted metamodels. *Computer  
 Methods in Applied Mechanics and Engineering*, 195(44):6312 – 6329, 2006. ISSN  
 0045-7825. doi: 10.1016/j.cma.2005.12.008.
- [455] [20] P. E. Gill, W. Murray, and M. A. Saunders. SNOPT: An SQP algorithm for large-  
 scale constrained optimization. *SIAM Review*, 47(1):99–131, 2005. doi: 10.1137/  
 S0036144504446096.
- [458] [21] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. MIT Press, 2016. URL  
<http://www.deeplearningbook.org>.
- [460] [22] J. S. Gray and J. R. R. A. Martins. Coupled aeropropulsive design optimization of a  
 boundary-layer ingestion propulsor. *The Aeronautical Journal*, 123(1259):121–137,  
 January 2019. doi: 10.1017/aer.2018.120.
- [463] [23] K. Gurney. *An Introduction to Neural Networks*. Taylor & Francis, Inc., Bristol, PA,  
 USA, 1997. ISBN 1857286731.

- [24] L. Haitao, W. Xiaofang, and X. Shengli. Generalized radial basis function-based high-dimensional model representation handling existing random data. *Journal of Mechanical Design*, 139(MD-16-1098):13, 2016. ISSN 1050-0472. doi: 10.1115/1.4034835.
- [25] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning*. Springer Series in Statistics. Springer New York Inc., New York, NY, USA, 2001.
- [26] S. Haykin. *Neural Networks: A Comprehensive Foundation*. Prentice Hall PTR, Upper Saddle River, NJ, USA, 2nd edition, 1998. ISBN 0132733501.
- [27] P. He, C. A. Mader, J. R. R. A. Martins, and K. J. Maki. An aerodynamic design optimization framework using a discrete adjoint approach with OpenFOAM. *Computers & Fluids*, 168:285–303, May 2018. doi: 10.1016/j.compfluid.2018.04.012.
- [28] S. He, E. Jonsson, C. A. Mader, and J. R. R. A. Martins. Aerodynamic shape optimization with time spectral flutter adjoint. In *2019 AIAA/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*, San Diego, CA, January 2019. American Institute of Aeronautics and Astronautics. doi: 10.2514/6.2019-0697.
- [29] X. He, J. Li, C. A. Mader, A. Yildirim, and J. R. R. A. Martins. Robust aerodynamic shape optimization—from a circle to an airfoil. *Aerospace Science and Technology*, 87:48–61, April 2019. doi: 10.1016/j.ast.2019.01.051.
- [30] G. Heigold, V. Vanhoucke, A. Senior, P. Nguyen, M. Ranzato, M. Devin, and J. Dean. Multilingual acoustic models using distributed deep neural networks. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Vancouver, CA, 2013. URL <https://static.googleusercontent.com/media/research.google.com/en//pubs/archive/40807.pdf>.
- [31] I. C. Kampolis, E. I. Karangelos, and K. C. Giannakoglou. Gradient-assisted radial basis function networks: theory and applications. *Applied Mathematical Modelling*, 28(2):197 – 209, 2004. ISSN 0307-904X. doi: 10.1016/j.apm.2003.08.002.
- [32] K. Kang, C. Qin, B. Lee, and I. Lee. Modified screening-based kriging method with cross validation and application to engineering design. *Applied Mathematical Modelling*, 70:626 – 642, 2019. ISSN 0307-904X. doi: 10.1016/j.apm.2019.01.030.
- [33] G. K. W. Kenway, C. A. Mader, P. He, and J. R. R. A. Martins. Effective adjoint approaches for computational fluid dynamics. *Progress in Aerospace Sciences*, 110: 100542, October 2019. doi: 10.1016/j.paerosci.2019.05.002.
- [34] L. Laurent, R. Le Riche, B. Soulier, and P.-A. Boucard. An overview of gradient-enhanced metamodels with applications. *Archives of Computational Methods in Engineering*, 26(1):61–106, January 2019. ISSN 1886-1784. doi: 10.1007/s11831-017-9226-3.
- [35] J. Li, M. A. Bouhlel, and J. R. R. A. Martins. Data-based approach for fast airfoil analysis and optimization. *AIAA Journal*, 57(2):581–596, February 2019. doi: 10.2514/1.J057129.

- 504 [36] R. P. Liem, G. K. W. Kenway, and J. R. R. A. Martins. Multimission aircraft fuel  
 505 burn minimization via multipoint aerostructural optimization. *AIAA Journal*, 53  
 506 (1):104–122, January 2015. doi: 10.2514/1.J052940.
- 507 [37] R. P. Liem, C. A. Mader, and J. R. R. A. Martins. Surrogate models and mix-  
 508 tures of experts in aerodynamic performance prediction for aircraft mission analysis.  
 509 *Aerospace Science and Technology*, 43:126–151, June 2015. doi: 10.1016/j.ast.2015.  
 510 02.019.
- 511 [38] W. Liu and S. Batill. Gradient-enhanced neural network response surface approx-  
 512 imations. *8th Symposium on Multidisciplinary Analysis and Optimization, Multi-  
 513 disciplinary Analysis Optimization Conferences, AIAA*, 2019. doi: doi:10.2514/6.  
 514 2000-4923.
- 515 [39] Z. Lyu, G. K. W. Kenway, and J. R. R. A. Martins. Aerodynamic shape optimization  
 516 investigations of the Common Research Model wing benchmark. *AIAA Journal*, 53  
 517 (4):968–985, April 2015. doi: 10.2514/1.J053318.
- 518 [40] M. Mangano and J. R. R. A. Martins. Multipoint aerodynamic shape opti-  
 519 mization for subsonic and supersonic regimes. In *57th AIAA Aerospace Sci-  
 520 ences Meeting, AIAA SciTech Forum, 2019*, San Diego, CA, January 2019. doi:  
 521 10.2514/6.2019-0696.
- 522 [41] J. R. R. A. Martins and J. T. Hwang. Review and unification of methods for comput-  
 523 ing derivatives of multidisciplinary computational models. *AIAA Journal*, 51(11):  
 524 2582–2599, November 2013. doi: 10.2514/1.J052184.
- 525 [42] J. R. R. A. Martins, P. Sturdza, and J. J. Alonso. The complex-step derivative ap-  
 526 proximation. *ACM Transactions on Mathematical Software*, 29(3):245–262, Septem-  
 527 ber 2003. doi: 10.1145/838250.838251.
- 528 [43] W. S. McCulloch and W. Pitts. A logical calculus of the ideas immanent in nervous  
 529 activity. *The bulletin of mathematical biophysics*, 5(4):115–133, Dec 1943. ISSN  
 530 1522-9602. doi: 10.1007/BF02478259.
- 531 [44] S. Pan and K. Duraisamy. Long-time predictive modeling of nonlinear dynamical  
 532 systems using neural networks. *Complexity*, 2018:1–26, December 2018. doi: 10.  
 533 1155/2018/4801012.
- 534 [45] N. Papila, W. Shyy, L. Griffin, and D. Dorney. Shape optimization of supersonic  
 535 turbines using response surface and neural network methods. In *39th Aerospace  
 536 Sciences Meeting and Exhibit*. American Institute of Aeronautics and Astronautics,  
 537 January 2001. doi: 10.2514/6.2001-1065.
- 538 [46] M. J. D. Powell. *The Theory of Radial Basis Function Approximation in 1990*, pages  
 539 105–210. Oxford University Press, May 1992.
- 540 [47] M. M. Rai and N. K. Madavan. Aerodynamic design using neural networks. *AIAA  
 541 Journal*, 38(1):173–182, January 2000. doi: 10.2514/2.938.

- 542 [48] J. Sacks, S. B. Schiller, and W. J. Welch. Designs for computer experiments. *Technometrics*, 31(1):41–47, 1989.
- 543
- 544 [49] N. R. Secco and J. R. R. A. Martins. RANS-based aerodynamic shape optimization  
545 of a strut-braced wing with overset meshes. *Journal of Aircraft*, 56(1):217–227,  
546 January 2019. doi: 10.2514/1.C034934.
- 547 [50] N. R. Secco, J. P. Jasa, G. K. W. Kenway, and J. R. R. A. Martins. Component-based  
548 geometry manipulation for aerodynamic shape optimization with overset meshes.  
549 *AIAA Journal*, 56(9):3667–3679, September 2018. doi: 10.2514/1.J056550.
- 550 [51] W. J. Welch, R. J. Buck, J. Sacks, H. P. Wynn, T. J. Mitchell, and M. D. Morris.  
551 Screening, predicting, and computer experiments. *Technometrics*, 34(1):15–25, 1992.  
552 doi: 10.1080/00401706.1992.10485229.
- 553 [52] H. G. Weller, G. Tabor, H. Jasak, and C. Fureby. A tensorial approach to computa-  
554 tional continuum mechanics using object-oriented techniques. *Computers in Physics*,  
555 12(6):620–631, 1998. doi: 10.1063/1.168744.
- 556 [53] Z. Xu, J. H. Saleh, and V. Yang. Optimization of supercritical airfoil design with  
557 buffet effect. *AIAA Journal*, pages 1–11, January 2019. doi: 10.2514/1.j057573.
- 558 [54] A. Yildirim, G. K. W. Kenway, C. A. Mader, and J. R. R. A. Martins. A Jacobian-  
559 free approximate Newton–Krylov startup strategy for RANS simulations. *Journal*  
560 *of Computational Physics*, 397:108741, Nov. 2019. ISSN 0021-9991. doi: 10.1016/j.  
561 jcp.2019.06.018.
- 562 [55] Y. Yu, Z. Lyu, Z. Xu, and J. R. R. A. Martins. On the influence of optimization  
563 algorithm and starting design on wing aerodynamic shape optimization. *Aerospace*  
564 *Science and Technology*, 75:183–199, April 2018. doi: 10.1016/j.ast.2018.01.016.
- 565 [56] Y. Zhang, W. J. Sung, and D. N. Mavris. Application of convolutional neural net-  
566 work to predict airfoil lift coefficient. In *2018 AIAA/ASCE/AHS/ASC Structures,*  
567 *Structural Dynamics, and Materials Conference*. American Institute of Aeronautics  
568 and Astronautics, 2018. doi: 10.2514/6.2018-1903.
- 569 [57] W. Zongmin. Hermite-birkhoff interpolation of scattered data by radial basis func-  
570 tions. *Approximation Theory and its Applications*, 8(2):1–10, Jun 1992. ISSN 1573-  
571 8175. doi: 10.1007/BF02836101.