Aerodynamic Shape Optimization Using Gradient-Enhanced Multifidelity Neural Networks

Jethro Nagawkar[*]
Iowa State University, Ames, Iowa, 50011, USA

Leifur Leifsson [†]
Purdue University, West Lafayette, Indiana, 47907, USA

Ping He [‡]
Iowa State University, Ames, Iowa, 50011, USA

In this work, the gradient-enhanced multifidelity neural networks (GEMFNN) algorithm is extended to handle multiple scalar outputs and applied to airfoil shape optimization. GEMFNN is a multifidelity variant of the gradient-enhanced neural networks (GENN) algorithm and uses both function and gradient information available at multiple levels of fidelity to yield accurate high-fidelity predictions. GEMFNN construction is similar to the multifidelity neural networks (MFNN) algorithm. GEMFNN is demonstrated on two test cases, a 20-variable analytical case, and benchmark case II developed by the AIAA Aerodynamic Design Optimization Discussion Group (ADODG). GEMFNN is compared to neural networks (NN), GENN, and MFNN in terms of the computational cost required to reach a global accuracy. This accuracy is measured using the coefficient of determination metric. For both cases, GEMFNN outperformed all the other ML algorithms to reach the target accuracy. For the 20-variable case, GEMFNN was eight times cheaper than GENN. For the airfoil optimization, GEMFNN was 0.3 hours (2%) cheaper than GENN. The resulting optimized airfoil had a 77.2 drag count (43.8%) difference compared to the baseline shape.

Nomenclature

A	=	airfoil cross-sectional area, [-]
$A_{baseline}$	=	baseline airfoil cross-sectional area, [-]
a	=	speed of sound in air, [m/s]
c	=	chord length, [-]
C_d	=	drag coefficient, $\frac{d}{q_\infty c}$, [-]
C_l	=	lift coefficient, $\frac{l}{q_\infty c}$, [-]
C_m	=	pitching moment coefficient, $\frac{m}{q_\infty c^2}$, [-]
C_p	=	pressure coefficient, $\frac{p-p_\infty}{q_\infty}$, [-]
ΔC_d	=	drag count, 1E-4
ΔC_l	=	lift count, 1E-2
d	=	drag force, [N]
l	=	lift force, [N]
$\mathbf{l}$	=	lower bounds of the design variables
m	=	pitching moment, [N/m]
M_∞	=	freestream Mach number, $\frac{V_\infty}{a}$, [-]
n_t	=	number of datasets
N	=	mini-batch size
N_t	=	number of testing points

[*]Ph.D. Candidate, Department of Aerospace Engineering, AIAA Student Member.
[†]Associate Professor, School of Aeronautics and Astronautics, AIAA Senior Member.
[‡]Assistant Professor, Department of Aerospace Engineering, AIAA Senior Member.

p	=	static pressure on airfoil surface, [N/m^2]
p_∞	=	freestream static pressure, [N/m^2]
q_∞	=	freestream dynamic pressure, $0.5\rho_\infty V_\infty^2$, [N/m^2]
R^2	=	Coefficient of determination, [-]
Re	=	Reynolds number, $\frac{\rho_\infty V_\infty c}{\mu_\infty}$, [-]
$\mathbf{u}$	=	upper bounds of the design variables
V_∞	=	freestream air velocity, [m/s]
$\mathbf{x}$	=	vector of design variables
$\mathbf{x}_\mathrm{L}$	=	low-fidelity model input/design variables
$\mathbf{x}_\mathrm{H}$	=	high-fidelity model input/design variables
$\mathbf{X}$	=	sampling plan vector
$\mathbf{y}$	=	observations of the sampling plan
$\mathbf{y}_\mathrm{L}$	=	low-fidelity model response
$\mathbf{y}_\mathrm{H}$	=	high-fidelity model response
$\mathbf{y}_t$	=	observations at testing point
$\nabla\mathbf{y}_\mathrm{L}$	=	gradient of low-fidelity model response with respect to inputs
$\nabla\mathbf{y}_\mathrm{H}$	=	gradient of high-fidelity model response with respect to inputs
$\hat{\mathbf{y}}_\mathrm{L}$	=	low-fidelity prediction
$\hat{\mathbf{y}}_\mathrm{H}$	=	high-fidelity prediction
$\hat{\mathbf{y}}_t$	=	ML responses at testing points
$\bar{y}_t$	=	mean of the testing prediction
y^+	=	non-dimensionalized first layer cell thickness, [-]
α	=	angle of attack, [deg]
β	=	learning rate hyperparameter
θ	=	parameters of the NN algorithms
μ_∞	=	freestream viscosity of air, [Ns/m^2]
μ_{R^2}	=	mean of R^2, [-]
ρ_∞	=	freestream air density, [kg/m^3]
σ_{R^2}	=	standard deviation of R^2, [-]
$\sigma_{threshold}$	=	global accuracy threshold

I. Introduction

Aerodynamic design optimization (ADO) is increasingly becoming an integral part of designing complex physical systems, such as aeroplanes, cars, trains, and wind turbines. ADO has traditionally relied on expensive high-fidelity simulations to calculate both the cost function and constraint values [1–3]. These methods typically require multiple and repetitive high-fidelity model evaluations during the iterative design process. When combined with a large number of design variables, the methods result in problems that can be difficult to solve in a reasonable time frame.

To overcome these challenges, several different surrogate modeling methods have been introduced. Surrogate modeling methods can be broadly classified into either data-fit methods [4] or multifidelity methods [5]. Data-fit methods include fitting a response surface through evaluated single-fidelity sample points. While multifidelity methods include using low-fidelity data along with a limited number of high-fidelity data to augment the predictive capabilities of single-fidelity surrogate models.

A variety data-fit methods exist in literature such as polynomial chaos expansions [6], Kriging [7] (also known as Gaussian process regression) and its variants [8–10], and support vector machines [11]. Kriging is the most widely used method in various engineering analysis and design tasks [12], amongst these methods. Kriging, however, suffers from a variety of issues such as being difficult to implement [13], being poor at approximating discontinuous functions [14], costly to use in the presence of a large number of data samples [12], and difficulty in handling high-dimensional problems [15]. To overcome these challenges, several different methods such as the use of gradient information [12, 16], and the partial least-squares correlation functions [9, 10], have been introduced. These methods, however, still suffer from some of the aforementioned important issues [13, 17].

Multifidelity methods use information from multiple levels of fidelities to enhance the prediction capabilities of surrogate models constructed from a limited number of high-fidelity data, while reducing the overall cost associated with constructing these surrogate models [5, 12]. Low-fidelity models are cheaper to evaluate, reducing the overall cost required in generating the data in order to construct the surrogate model. Cokriging [18], which is the multifidelity version of Kriging, and its variants [19, 20], are becoming popular in design and analysis tasks. Cokriging, however, still suffers from the same issues associated with Kriging [13].

Neural networks (NN) [21] are becoming more prevalent in engineering design and analysis problems [13, 17]. NN can handle high-dimensional datasets [17], are scalable with the number of data [21], are easy to implement [13], and can approximate discontinuous data [13]. The biggest drawback of NN is that they require a large number of samples to make accurate predictions [21], especially for high-dimensional problems [17]. Methods such as gradient-enhanced NN (GENN) [17, 22], and multifidelity NN (MFNN) [13], have been recently introduced, to overcome these challenges.

In this work, the gradient-enhanced multifidelity NN (GEMFNN) algorithm proposed by Nagawkar and Leifsson [23] is extended to handle multiple outputs and is applied to aerodynamic shape optimization. GEMFNN is a multifidelity variant of GENN [22] and its construction is similar to MFNN. GEMFNN leverages both function and gradient information available from low- and high-fidelity models to yield accurate high-fidelity predictions. In this work, the GEMFNN algorithm is demonstrated on two problems, one analytical problem and the AIAA Aerodynamic Design Optimization Discussion Group (ADODG) benchmark case II problem. Several works on using different computational fluid dynamics (CFD) solvers, shape parameterizations, and optimization algorithms have been reported for ADODG case II [24–31]

The next section describes the GEMFNN algorithm. In the following section, this machine learning (ML) algorithm is applied to two different test cases and compared to other similar ML algorithms. Lastly, the conclusions and future work are presented.

II. Methods

The construction of the GEMFNN ML algorithm is described in this section. First, an outline of the GEMFNN-based analysis is introduced, followed by the sampling plan used to generate the training and testing data. The construction of the GEMFNN algorithm is then discussed, followed by the validation metric to quantify its global accuracy. The final section describes the application of the GEMFNN algorithm to aerodynamic design optimization.

A. Workflow

Figure 1 shows a flowchart of the GEMFNN algorithm. The design space is first sampled in order to generate the data to train and test the GEMFNN model. Both the high- and low-fidelity models are evaluated along with their gradients to generate the responses in order to train the ML model. Once trained, this model is tested using the testing data. The coefficient of determination (R^2) error metric is used for validation. If the trained model has a R^2 value below $\sigma_{threshold}$, the previous steps are repeated with an increased number of samples in the training dataset. If R^2 is above $\sigma_{threshold}$, the GEMFNN model is sufficiently accurate to be used for aerodynamic shape optimization. The choice of $\sigma_{threshold}$ is case dependent.

B. Sampling plan

Sampling is the first set involved in constructing the ML algorithms. It is the process of selecting discrete samples in the variable space [12]. The Latin Hypercube sampling (LHS) [32] plan is used in this work to generate both the high- and low-fidelity training data as well as the testing data.

C. Gradient-Enhanced Multifidelity Neural Networks

NN are universal function approximators [21]. It contains a hierarchy of features, known as layers. Hidden layers are layers in-between the input and output layers. The output and each hidden layer contain neurons, which are a fundamental unit of computation. They contain an activation function [21]. An unconstrained optimization problem is solved in NN, where the parameters, θ, of the NN are tuned using a gradient-based optimizer [21]. In this study, the Adaptive Moments (ADAM) [33] optimization algorithm is used. The backpropagation algorithm [34] is used to compute the gradients for the gradient-based optimizer. The mean squared error (MSE) is used as the loss function in the optimization problem, in this study, and is given by

$$\mathcal{L}_{NN} = \frac{\sum_{l=1}^{N} (\hat{y}_{\mathrm{H}}^{(l)} - y_{\mathrm{H}}^{(l)})^2}{N}, \tag{1}$$

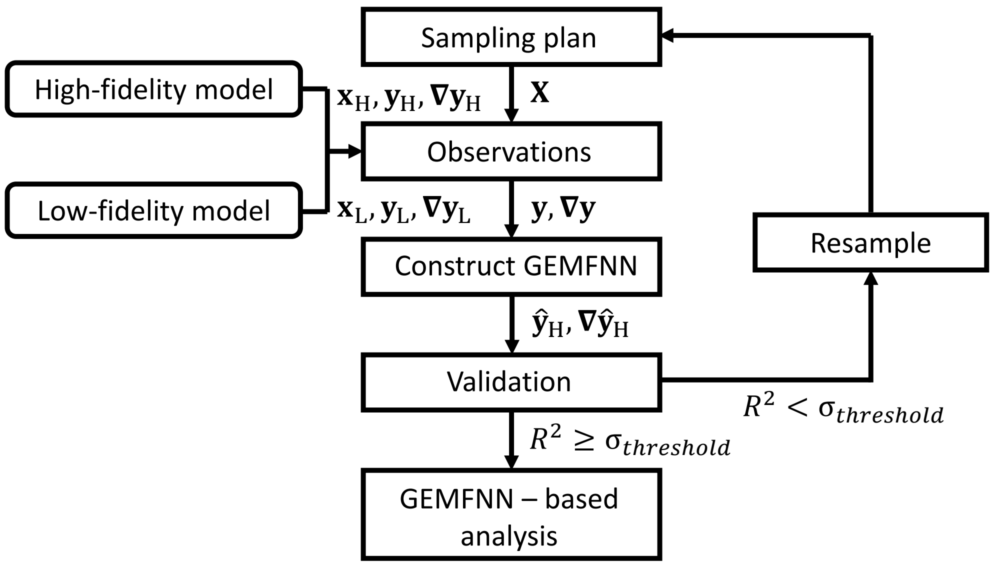

Sampling plan
High-fidelity model
$\mathbf{x}_\mathrm{H}, \mathbf{y}_\mathrm{H}, \nabla\mathbf{y}_\mathrm{H}$
$\mathbf{X}$
Observations
Low-fidelity model
$\mathbf{x}_\mathrm{L}, \mathbf{y}_\mathrm{L}, \nabla\mathbf{y}_\mathrm{L}$
$\mathbf{y}, \nabla\mathbf{y}$
Resample
Construct GEMFNN
$\hat{\mathbf{y}}_\mathrm{H}, \nabla\hat{\mathbf{y}}_\mathrm{H}$
Validation
$R^2 < \sigma_{threshold}$
$R^2 \geq \sigma_{threshold}$
GEMFNN – based analysis

Fig. 1 Flowchart of the gradient-enhanced multifidelity neural network algorithm.

where N is the number of samples in a subset of the training data, called mini-batch [21]. It is used to minimize the mismatch between the high-fidelity training data observations, y_H, and the predicted values, $\hat{y}_\mathrm{H}$, of the NN.

In GENN [17], the loss function in (1) is modified by adding the mismatch match between the high-fidelity training data gradient, ∇y_H, and the predicted gradient of the NN, $\nabla\hat{y}_\mathrm{H}$, to it, and is given by

$$\mathcal{L}_{GENN} = \mathcal{L}_{NN} + \frac{\sum_{l=1}^{N}\sum_{k=1}^{D}(\nabla\hat{y}_{\mathrm{H},k}^{(l)} - \nabla y_{\mathrm{H},k}^{(l)})^2}{N}, \tag{2}$$

where D is the input variable space dimension. This loss function minimizes the mismatch between both the function and its tangent at a given training point to the corresponding true values, respectively.

Figure 2 shows a schematic of the MFNN [13] architecture. It contains three NN. NN_L is used to approximate the low-fidelity data. Its output is used as an additional input variable to two other NN, NN_{H_1} and NN_{H_2}. NN_{H_1} captures linear ($\tilde{y}_l$) correlations between high- and low-fidelity data, while NN_{H_2} captures nonlinear ($\tilde{y}_{nl}$) correlations. The weighted sum of the outputs of the linear and nonlinear layer gives the high-fidelity prediction of the MFNN as

$$\hat{y}_\mathrm{H} = \omega\tilde{y}_l + (1-\omega)\tilde{y}_{nl}, \tag{3}$$

where ω is an additional parameter of the MFNN, which is tuned along with the other parameters of the MFNN using a gradient-based optimizer. It is important to note that NN and GENN are single-fidelity models and use only the NN_{H_2} part of MFNN. They do not include $\hat{y}_\mathrm{L}$ as an additional input parameter since they directly make high-fidelity predictions. The loss function of MFNN is given as

$$\mathcal{L}_{MFNN} = \mathcal{L}_{NN} + \frac{\sum_{l=1}^{N}(\hat{y}_\mathrm{L}^{(l)} - y_\mathrm{L}^{(l)})^2}{N}. \tag{4}$$

The proposed GEMFNN algorithm is a novel multifidelity version of GENN [22] and is constructed similar to MFNN [13]. It uses gradient information available at high- and low-fidelity data during training. The new and unique loss function for GEMFNN is taken as

$$\begin{aligned}\mathcal{L}_{GEMFNN} =& \mathcal{L}_{MFNN} + \frac{\sum_{l=1}^{N}\sum_{k=1}^{D}(\nabla\hat{y}_{\mathrm{L},k}^{(l)} - \nabla y_{\mathrm{L},k}^{(l)})^2}{N} \\ &+ \frac{\sum_{l=1}^{N}\sum_{k=1}^{D}(\nabla\hat{y}_{\mathrm{H},k}^{(l)} - \nabla y_{\mathrm{H},k}^{(l)})^2}{N}.\end{aligned} \tag{5}$$

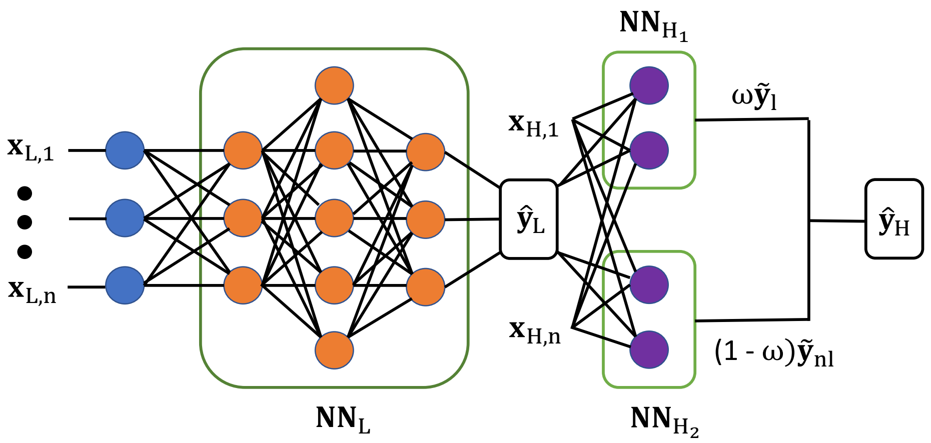

$\mathbf{NN}_{\mathrm{H}_1}$
$\omega\tilde{\mathbf{y}}_{\mathrm{l}}$
$\mathbf{x}_{\mathrm{L},1}$
$\mathbf{x}_{\mathrm{H},1}$
$\hat{\mathbf{y}}_{\mathrm{L}}$
$\hat{\mathbf{y}}_{\mathrm{H}}$
$\mathbf{x}_{\mathrm{L,n}}$
$\mathbf{x}_{\mathrm{H,n}}$
$(1 - \omega)\tilde{\mathbf{y}}_{\mathrm{nl}}$
$\mathbf{NN}_{\mathrm{L}}$
$\mathbf{NN}_{\mathrm{H}_2}$

Fig. 2 Multifidelity neural network architecture (adapted from [13]).

The steps in the GEMFNN algorithm are the following:

1) Normalize the input, output and gradient of output with respect the inputs for all the data.
2) Perform forward propagation through NN_L to get $\hat{y}_{\mathrm{L}}$ and through MFNN to get $\hat{y}_{\mathrm{H}}$.
3) Calculate $\nabla\hat{y}_{\mathrm{L}}$ and $\nabla\hat{y}_{\mathrm{H}}$ using reverse mode automatic differentiation [35].
4) Use (5) to calculate $\mathcal{L}_{GEMFNN}$.
5) Calculate the gradient of $\mathcal{L}_{GEMFNN}$ with respect to all the parameters in GEMFNN ($\nabla\mathcal{L}_{GEMFNN}$) using the backpropagation [34] algorithm.
6) Update the parameters of the MFNN model:

$$\theta \leftarrow \theta - \beta\nabla\mathcal{L}_{GEMFNN}, \tag{6}$$

   where β is the learning rate hyperparameter.
7) Iterate over steps $2 - 6$ till all the mini-batches present in one epoch is used. One epoch is one iteration over an entire training dataset [21].
8) Repeat steps $2 - 7$ for all the epochs.
9) GEMFNN is now trained and ready to be used for aerodynamic shape optimization.

D. Validation

The coefficient of determination is used to measure the global accuracy of the ML model in this work, and is given as

$$R^2 = 1 - \frac{\sum_{j=1}^{N_t}(y_t^{(j)} - \hat{y}_t^{(j)})^2}{\sum_{j=1}^{N_t}(y_t^{(j)} - \bar{y}_t)^2}, \tag{7}$$

where $\hat{y}_t^{(j)}$ and $y_t^{(j)}$ are the ML model estimation and high-fidelity observation of the j^{th} testing point, respectively, N_t is the total number of testing data samples in one dataset, and $\bar{y}_t$ is the mean of $y_t^{(j)}$, given by

$$\bar{y}_t = \frac{\sum_{j=1}^{N_t} y_t^{(j)}}{N_t}. \tag{8}$$

R^2 is the measure of "Goodness of fit" [12] of an algorithm. A R^2 less than zero implies that the mean of the true function is better than the algorithm at predicting the trend of the true function. $R^2 = 0$ implies that the algorithm prediction is the same as the mean of the true function. If $R^2 = 1$, the algorithm has approximated the true function perfectly. This makes it easier to chose the global accuracy criterion. In this work, a value of R^2 greater than $\sigma_{threshold}$ is considered an acceptable global accuracy. The choice of $\sigma_{threshold}$ is case dependent and its values can be found under each case in Sec. III.

In order to account for the variation in the training data used, each ML model is retrained with 'n_t' different datasets. Both the mean and standard deviations are plotted to quantify the effect of using different datasets. The mean of R^2 is given by

$$\mu_{R^2} = \frac{\sum_{k=1}^{n_t} R_k^2}{n_t}, \tag{9}$$

and the standard deviation is given by

$$\sigma_{R^2} = \sqrt{\frac{\sum_{k=1}^{n_t} (R_k^2 - \mu_{R^2})^2}{n_t}}. \tag{10}$$

In this work, n_t is set to ten for the analytical case. For the airfoil optimization, n_t is set to one due to high computational requirements involved in obtaining the data.

E. GEMFNN-based analysis

In this study, the GEMFNN algorithm is used as a surrogate model to prediction the lift, drag, and pitching moment coefficients of an airfoil parameterized using free-form deformation (FFD) [36]. The FFD design variables, as well as the angle of attack, are the inputs to the GEMFNN algorithm and the aerodynamic coefficients are the outputs. Once this algorithm is trained with sufficient high-fidelity data to meet the global accuracy threshold, it is used along with the sequential least squares programming (SLSQP) [37] gradient-based optimizer to solve the ADODG benchmark case II problem.

III. Numerical examples

The GEMFNN ML algorithm is demonstrated on two different cases. The first case is a 20-variable analytical function and the second is the benchmark case II developed by ADODG. The GEMFNN ML algorithm is compared to other ML algorithms, namely, NN, GENN, and MFNN.

A. Case 1: 20-dimensional analytical function

The 20-dimensional high-fidelity analytical function [13] is written as

$$f_{\text{HF}}(\mathbf{x}) = (x_1 - 1)^2 + \sum_{i=2}^{20} (2x_i^2 - x_{i-1})^2, \tag{11}$$

where $\mathbf{x} \in$ [-3,3]. The corresponding high-fidelity gradient is

$$\nabla f_{\text{HF},1}(\mathbf{x}) = 2(x_1 - 1) - 2(2x_2^2 - x_1), \tag{12}$$

$$\nabla f_{\text{HF},i}(\mathbf{x}) = 8x_i(2x_i^2 - x_{i-1}) - 2(2x_{i+1}^2 - x_i), \tag{13}$$

for $i = 2, 3, ..., 19$, and

$$\nabla f_{\text{HF},20}(\mathbf{x}) = 8x_{20}(2x_{20}^2 - x_{19}). \tag{14}$$

The low-fidelity model is written as [13]

$$f_{\text{LF}}(\mathbf{x}) = 0.8 f_{\text{HF}}(\mathbf{x}) - \sum_{i=1}^{19} 0.4 x_i x_{i+1} - 50, \tag{15}$$

while the corresponding low-fidelity gradient is written as

$$\nabla f_{\text{LF},1}(\mathbf{x}) = 0.8 \nabla f_{\text{HF},1}(\mathbf{x}) - 0.4x_2, \tag{16}$$

$$\nabla f_{\text{LF},i}(\mathbf{x}) = 0.8 \nabla f_{\text{HF},i}(\mathbf{x}) - 0.4(x_{i-1} + x_{i+1}), \tag{17}$$

for $i = 2, 3, ..., 19$, and

$$\nabla f_{\text{LF},20}(\mathbf{x}) = 0.8 \nabla f_{\text{HF},20}(\mathbf{x}) - 0.4x_{19}. \tag{18}$$

The LHS plan was used to generate the training and testing datasets. The testing data contains $10,000$ samples. NN_{H_1} contains one hidden layer with ten neurons, while NN_{H_2} contains four hidden layers, with 64 neurons each, and NN_L contains six hidden layers with 128 neurons each.

The tangent hyperbolic activation function is used in NN_{H_2} and NN_L. The batch size is fixed to a value of 64, while the learning rate is set to 0.001, and the number of epochs is fixed with a value of $10,000$. No regularization is used in this case. This setup is used for all the four ML algorithms.

The high-fidelity sampling cost required by each ML algorithm to reach $\sigma_{threshold}$, is shown in Table 1. $\sigma_{threshold}$ for this case is set to a value of 0.99 R^2. GEMFNN required the least cost to reach this accuracy. GEMFNN required 600 samples (300 function plus 300 gradient), while GENN required $5,000$ samples ($2,500$ function plus $2,500$ gradient). NN and MFNN fail to meet the target accuracy, even with $10,000$ samples. Figure 3(a) shows the number of high-fidelity samples required by each algorithm to meeting the target accuracy. This plot is generated by averaging the results over ten different datasets. Figure 3(b) shows the corresponding standard deviations of the algorithms. Increasing the number of samples, decreases the standard deviations, resulting in algorithms that are less sensitive the the training data. This case demonstrates the benefit of using both gradient and multifidelity information for a high-dimensional problem.

B. Case 2: ADODG benchmark case II

1. Problem formulation

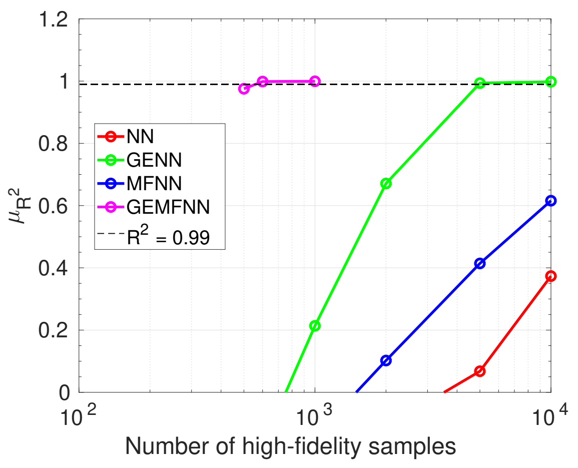
In this case, the drag coefficient (C_d) of the RAE 2822 airfoil in viscous flow of freestream Mach number (M_∞) equal to 0.734, and a Reynolds number of 6.5×10^6, is minimized. This airfoil is subject to a fixed lift coefficient (C_l) of 0.824

Table 1 20-dimensional function modeling cost.

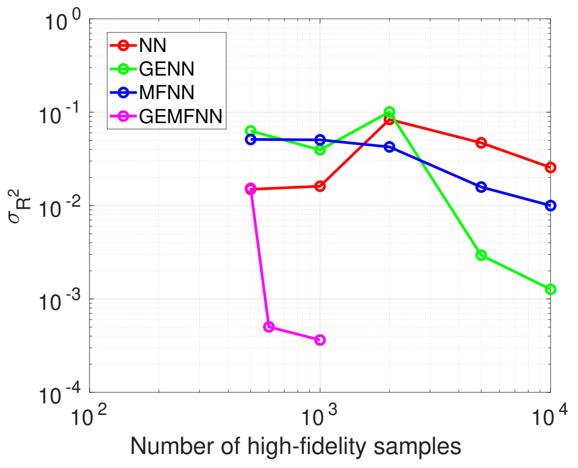
ML algorithm	High-fidelity sample cost
NN	>10,000
GENN	5,000*
MFNN	>10,000**
GEMFNN	600*,**

*Function plus gradient evaluation cost

**Plus 30,000 low-fidelity training samples



(a)



(b)

Fig. 3 20-dimensional function results: (a) mean of R^2, (b) standard deviation of R^2.

as well as constraints on the pitching moment coefficient (C_m) and the area constraints.

The optimization problem is stated as:

$$\min_{\mathbf{l} \le \mathbf{x} \le \mathbf{u}} C_d(\mathbf{x}) \tag{19}$$

subject to the equality constraint

$$C_l(\mathbf{x}) = 0.824 \tag{20}$$

and inequality constraints

$$C_m(\mathbf{x}) \ge -0.092 \tag{21}$$

and

$$A(\mathbf{x}) \ge A_{baseline}. \tag{22}$$

$\mathbf{x}$ is the design variable vector, while $\mathbf{l}$ and $\mathbf{u}$ are the lower and upper bounds, respectively, of each design variable. A is the cross-sectional area of the airfoil non-dimensionalised with the square of the chord length (c). $A_{baseline}$ is the area of the RAE 2822 airfoil.

2. *Shape parameterization*

Free-form deformation (FFD) [36], implemented by Kenway et al. [38], is used to parameterize the airfoil geometry and is shown in Fig. 4. The FFD approach embeds the geometry into a volume that can then be manipulated by moving points at the surface of that volume (the FFD points). Once an object is embedded into a FFD volume, a Newton search is executed to determine the mapping between the FFD points (parameter space) and the surface geometry (physical space). The FFD volume is a trivariate B-spline volume such that the gradient of any point inside the volume can be easily computed. The geometry parameterization is implemented in the pyGeo [38] module and allows for control of the local shape of the geometry during optimization. In this study, a total of twelve FFD control points, six for the upper surface and six for the lower, are used to parameterize the airfoil geometry, with locations shown in Fig. 4. The angle of attack along with the FFD control points form the design variables for this case.

3. *CFD modeling*

To simulate the flow and compute the gradients, the finite-volume structured multiblock mesh solver ADflow [39], is used. ADflow solves the compressible Reynolds-averaged Navier-Stokes (RANS) equations simultaneously. The Spalart-Allmaras [40] turbulence model is used for this case, which ADflow solves in a segregated manner. An approximate Newton-Krylov (ANK) [41] and a full Newton-Krylov (NK) approaches are used to converge the residuals. The ANK solver is robust to act as a globalization scheme for the full NK solver and converges well even without multigrid. The NK solver provides efficient terminal convergence for all types of meshes once the solution is within the Newton basin of attraction. ADflow uses a Jacobian-free discrete adjoint method [42] for gradient computation, which is machine-precision accurate.

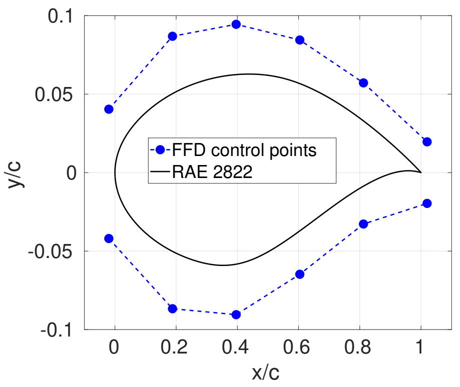
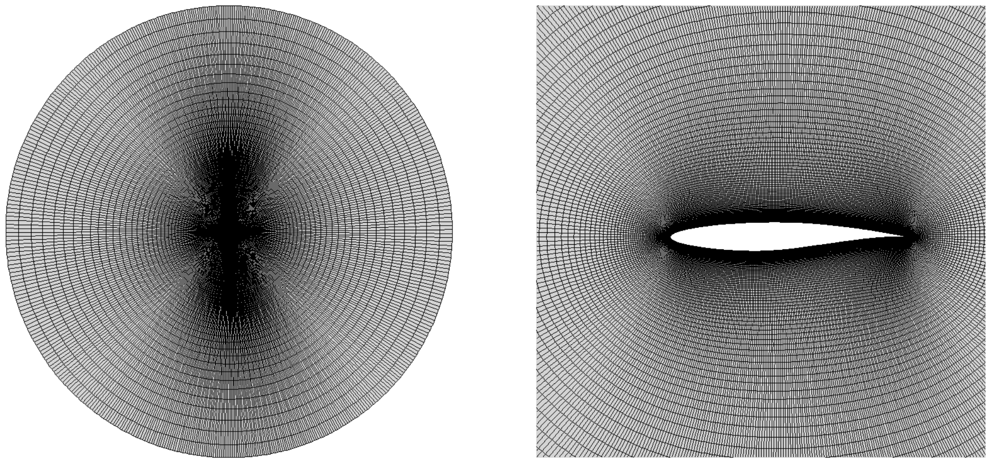


Fig. 4 Free-form deformation parameterization of the airfoil surface.


(a)

(b)

Fig. 5 O-grid for Case 2: (a) far-field, (b) surface.

An O-grid mesh, generated using pyHyp [43], is used for this case and is shown in Fig. 5. The far-field is set to a distance of 100c from the airfoil. The y^+ is set to a value of less than one. The far-field Mach number is set to a value of 0.734, and the Reynolds number is set to a value of 6.5×10^6. A fixed angle of attack is used for the CFD simulations. The convergence criteria for the flow residual norm, relative to its initial value, is set to a value of 10^{-14}, and for the adjoint residual norm, it is set to a value of 10^{-12}.

A grid convergence study for the RAE 2822 airfoil is performed and shown in Table 2. This study is performed for a fixed lift coefficient of 82.4 lift counts and the corresponding angle of attack, drag and pitch moment coefficients are shown in Table 2. Mesh L1 is used as the high-fidelity model and Mesh L3 as the low-fidelity model. To calculate the cost of the high- and low-fidelity flow and adjoint time, ten different airfoil shapes were generated using LHS, for different angles of attack. The high- and low-fidelity flow time calculated was 140.6 and 9.0 seconds, respectively. The low-fidelity model flow time is therefore 15.6 times cheaper than the high-fidelity model. The corresponding high- and low-fidelity adjoint time was 101.9 and 10.2 seconds, respectively, making the low-fidelity adjoint time an order of magnitude cheaper than the high-fidelity model. Note that the reported adjoint time is the total adjoint simulation time for the three aerodynamic coefficients, namely, the drag, lift, and pitching moment. The Mach number contour for the high-fidelity model is shown in Fig. 6 (a). The shock wave can be seen where the contour changes color from red to green. Figure 6 (b) shown the differences in the pressure coefficient profile for the high- and low-fidelity models.

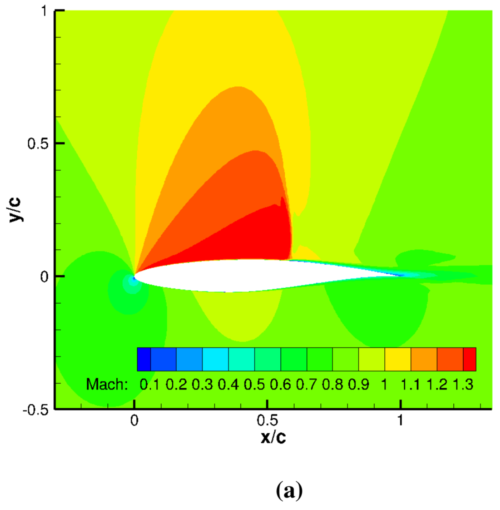
4. Results

The variation of the R^2 error metric with respect to the number of high-fidelity samples and computational cost is shown in Fig. 7 for the different ML algorithms. To calculate this error metric, a separate testing set containing flow simulation results from 845 different airfoil shapes, is used. Throughout this case, the testing set used is kept constant.

Table 2 Grid convergence study of the baseline shape for Case 2.

Mesh	Number of cells	α (deg)	C_d (d.c.)	$C_{m,c/4}$	Flow time* (s)	Adjoint time*,** (s)
L3	8,000	2.55	186.6	-0.1088	7.9	9.6
L2	32,000	2.68	180.6	-0.1015	19.4	22.7
L1	128,000	2.70	177.1	-0.1000	127.9	95.3
L0	320,000	2.68	176.2	-0.1004	657.3	332.5

*Computed on a high-performance cluster with 16 processors.
**Total time for three adjoint simulations.



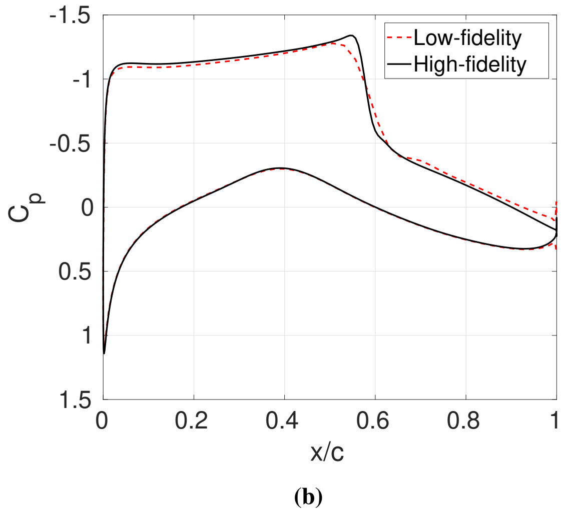
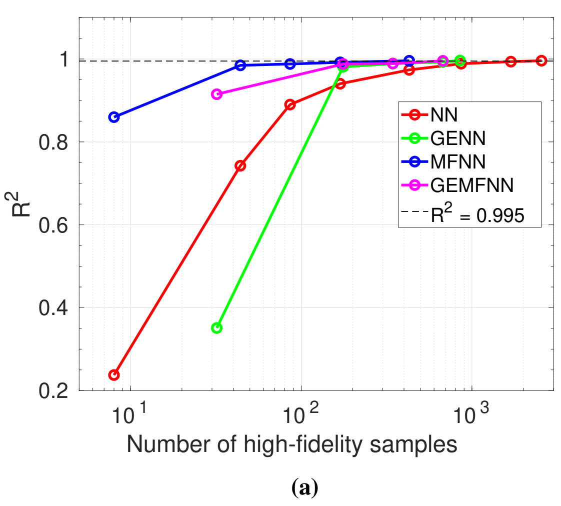


Fig. 6 CFD results for RAE 2822 airfoil: (a) Mach number contours (mesh L1), (b) pressure coefficient profile (low-fidelity: mesh L3, high-fidelity: mesh L1).



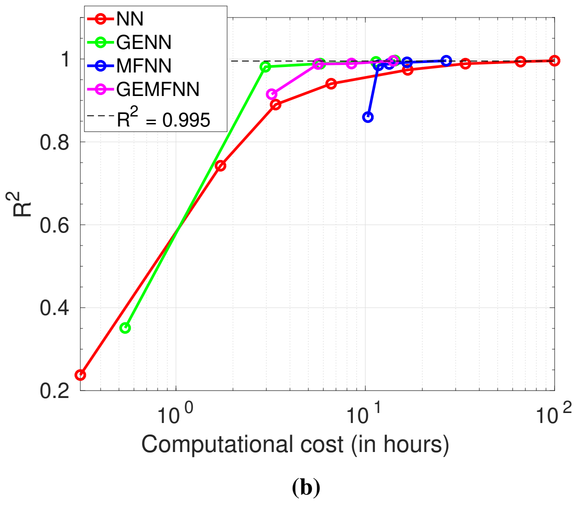


Fig. 7 R^2 convergence history with respect to : (a) number of high-fidelity samples, (b) computing time, for all ML algorithms.

A hyperparameter sweep was performed for all the ML algorithms and at each high-fidelity sample size used to train the algorithms. The combination of hyperparameters which resulted in the highest global accuracy for each ML algorithm and for each high-fidelity sample size was used to report the R^2 values in Figs. 7, 8, and 9. These hyperparameters varied between each ML algorithm and for each high-fidelity sample size. The hyperparameters tuned included the number of hidden layers, the number of neurons in each hidden layer, the activation function, the batch size, the total number of epochs, the learning rate, the type a regularization used, and the value of the regularization coefficient.

Figure 7 (a) shows that MFNN requires the least number of high-fidelity samples to reach the target accuracy of 0.995 on the R^2 metric. Note that, for the gradient-enhanced cases, number of high-and low-fidelity sample cost includes the cost of obtaining both the function and the gradients. In terms of computational cost required to obtain the target accuracy, GEMFNN results the least time, just 0.3 hours lower than GENN. The cost of all the ML algorithms is summarized in Table 3. To find the ideal number of low-fidelity samples for the multifidelity cases, the R^2 metric convergence is studied for different number of low-fidelity samples. The convergence for MFNN and GEMFNN are

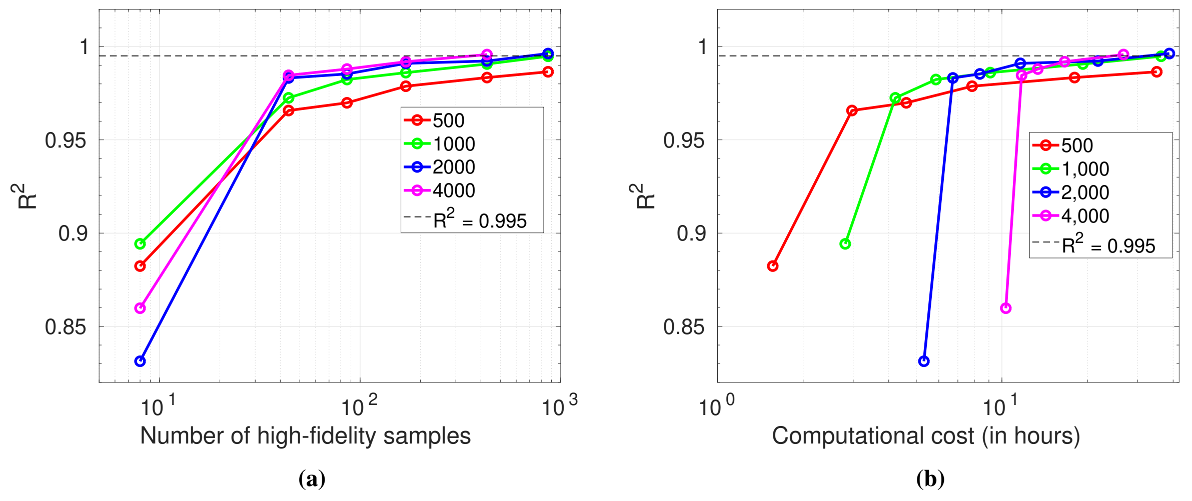


Fig. 8 MFNN R^2 convergence history with respect to : (a) number of high-fidelity samples, (b) computing time, for different number of low-fidelity samples.

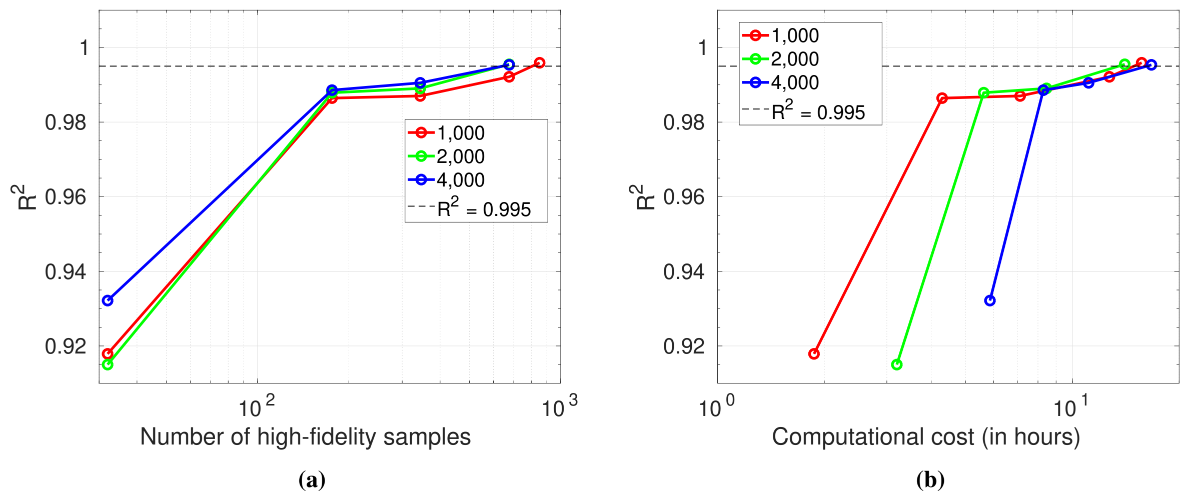


Fig. 9 GEMFNN R^2 convergence history with respect to : (a) number of high-fidelity samples, (b) computing time, for different number of low-fidelity samples.

shown in Figs. 8 and 9, respectively. MFNN reaches the target accuracy fastest with 4,000 low-fidelity samples, while GEMFNN with 2,000.

Once the target accuracy is met, the ML algorithms are used as surrogate models to find the optimum airfoil shapes. On obtaining the optimum shapes, these airfoils are simulated in ADflow to obtain the objective and constraint values at a lift coefficient of 82.4 lift counts. The Mach number contours for the optimum shapes obtained using the different ML algorithms are shown in Fig. 10. The Mach number counters are look similar for the different ML algorithms, however, small variations can be seen around the airfoil surface. Figure 11 shows the optimized airfoil shapes and pressure coefficient profiles obtained from the different ML algorithms compared to the RAE 2822 airfoil. The shock strength has drastically reduced and the results are nearly shock-free. Table 4 reports the results of the optimized airfoil shapes. GENN has the lowest drag of 97.6 drag counts, while GEMFNN has a slightly higher drag of 97.8 drag counts. Work done by He et al. [24] suggests that the results for this benchmark case has a global optimum with a shock-free solution. A shock-free solution is not obtained in this study. While the global accuracy of the ML algorithm is high, the local accuracy at the global optimum is not high enough to capture the shock-free solution. Further work needs to be done to

Table 3 Case 2 modeling cost.

ML algorithm	High-fidelity sample cost	Low-fidelity sample cost	Computational cost (h)
NN	2,557	-	99.9
GENN*	852	-	14.4
MFNN	429	4,000	26.8
GEMFNN*	676	2,000	14.1

*Flow plus adjoint evaluation cost

address this issue. Table 5 shows the grid convergence study performed on the optimized airfoil shape, obtained using

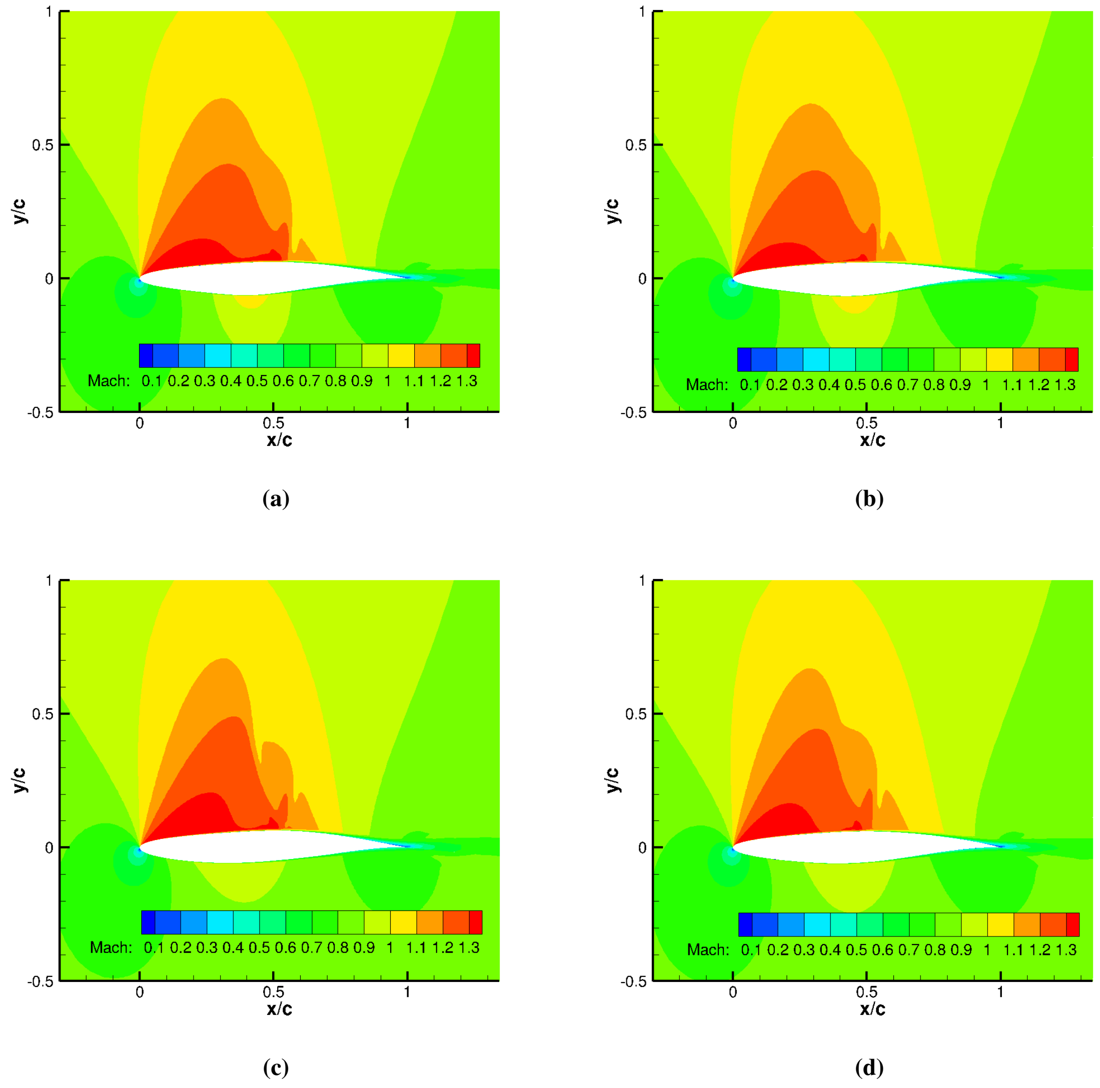


Fig. 10 Mach number contours of the optimized designs using various surrogates: (a) NN, (b) GENN, (c) MFNN, (d) GEMFNN.

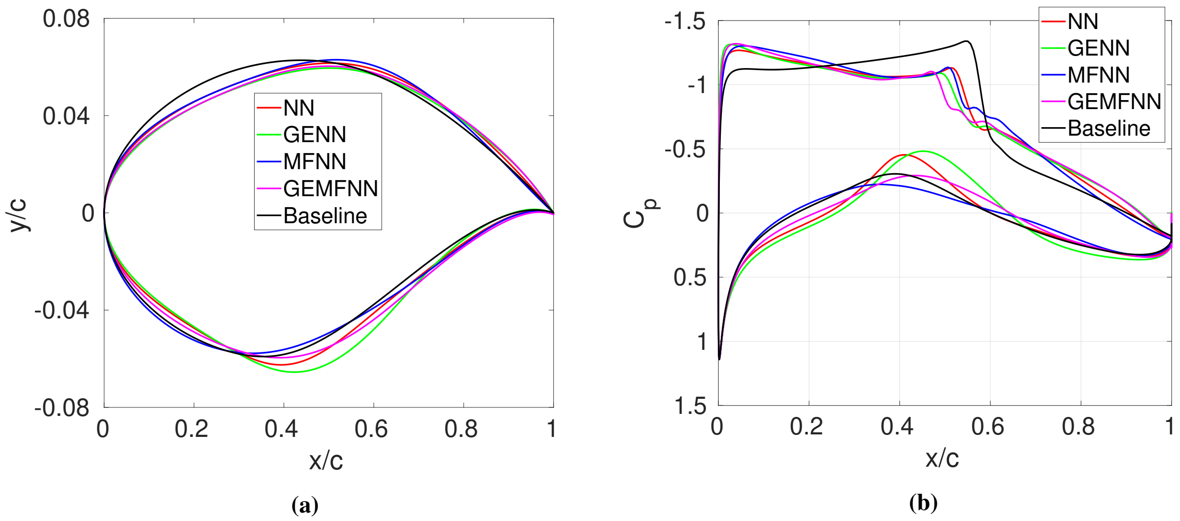


Fig. 11 Case 2 baseline vs optimized results: (a) airfoil shape, (b) pressure coefficient profile.

Table 4 Optimization results for Case 2.

Parameter/Method	Baseline	NN	GENN	MFNN	GEMFNN
C_d (d.c.)	177.1	99.0	97.6	100.2	97.8
$C_{m,c/4}$	-0.1000	-0.0928	-0.0926	-0.091	-0.0930
A	0.07767	0.07768	0.07768	0.07768	0.07768
α (deg)	2.70	2.64	2.68	2.69	2.65

Table 5 Grid convergence study of the optimized shape (using GEMFNN) for Case 2.

Mesh	Number of cells	α (deg)	C_d (d.c.)	$C_{m,c/4}$
L3	8,000	2.37	113.4	-0.1091
L2	32,000	2.52	99.1	-0.0993
L1	128,000	2.65	97.8	-0.0930
L0	320,000	2.68	99.0	-0.0918

GEMFNN algorithm. A total drag reduction of 77.2 drag counts is obtained using the GEMFNN algorithm.

IV. Conclusion

A gradient-enhanced multifidelity neural network (GEMFNN) algorithm is proposed for aerodynamic shape optimization. The construction of GEMFNN is similar to multifidelity neural networks (MFNN) and it is a multifidelity extension of gradient-enhanced neural networks (GENN). In addition to the mean squared error (MSE) loss function used by NN in this study, GENN uses the MSE of the true and predicted gradient values. GEMFNN consists of three NN, one to approximate the low-fidelity data (NN_L), which is then connected to two other NN, one with linear (NN_{H_1}) and the other with nonlinear (NN_{H_2}) activation functions. This helps capture both linear and nonlinear correlations between the high- and low-fidelity data.

GEMFNN is demonstrated on two cases, namely, a 20-dimensional analytical problem, as well as benchmark case II developed by the AIAA Aerodynamic Design Optimization Discussion Group (ADODG). GEMFNN is compared to NN, GENN, and MFNN. For both cases, the variation of the coefficient of determination (R^2) metric with the number of high-fidelity samples is studied. GEMFNN is the most computationally efficient for both cases. For the analytical case, it is eight times cheaper than GENN. The benefit of using both gradient and multifidelity information in training the ML algorithms for the 20-dimensional cases is shown for this case. For the the airfoil case, however, it is just 0.3 hours cheaper than GENN. In terms of high-fidelity sampling cost, GEMFNN required 176 fewer samples than GENN. Further cost reduction for the multifidelity ML algorithms can be achieved by using a different low-fidelity model, such as a coarse mesh with lower convergence criteria for the residuals, or by using a separate surrogate model in-place of CFD. This, however, needs to be investigated. In this study, only twelve FFD points are used to parameterize the airfoil. Future work will involve studying the effect of the number of design variables on the high- and low-fidelity sampling cost and the resulting optimum shape.

Once the global target accuracy of the ML algorithm is met, it is used as a surrogate model to find the optimum design. The resulting optimized shapes from all the ML algorithms had nearly shock-free solution. Small variations in the shape of the optimized airfoil can be seen, which resulted in different shock strengths and drag coefficient values. The optimum design using GEMFNN had a 77.2 drag count difference from the baseline airfoil shape.

While GEMFNN was the most cost efficient method to reach the target accuracy for the airfoil optimization case, it was only slightly cheaper than GENN. However, what is noticeable, is that, in the absence of a large number of high-fidelity data, the multifidelity ML algorithms were significantly more accurate than the single-fidelity ones. If a method to quantify uncertainty in these algorithms is developed, it is possible to perform efficient global optimization (EGO) starting with a small number of high-fidelity samples. This could also overcome the problem in local inaccuracy while finding the optimum shape and could potentially find the shock-free global optimum. Both the uncertainty quantification in ML as well as using EGO with these algorithms will be done in future work.

Acknowledgment

This material is based upon work supported in part by the National Science Foundation under grant number 1846862.

References

[1] Skinner, S., and Zare-Behtash, H., "State-of-the-art in aerodynamic shape optimisation methods," *Applied Soft Computing*, Vol. 62, 2018, pp. 933 – 962. doi:10.1016/j.asoc.2017.09.030.

[2] Mader, C. A., and Martins, J. R. R. A., "Derivatives for Time-Spectral Computational Fluid Dynamics Using an Automatic Differentiation Adjoint," *AIAA Journal*, Vol. 50, 2012, pp. 2809–2819. doi:10.2514/1.J051658.

[3] Leung, T. M., and Zingg, D. W., "Aerodynamic shape optimization of wings using a parallel newton-krylov approach," *AIAA Journal*, Vol. 50, No. 3, 2012, pp. 540–550.

[4] Queipo, N. V., Haftka, R. T., Shyy, W., Goel, T., Vaidyanathan, R., and Tucker, P. K., "Surrogate-based analysis and optimization," *Progress in Aerospace Sciences*, Vol. 21, No. 1, 2005, pp. 1–28.

[5] Peherstorfer, B., Wilcox, K., and Gunzburger, M., "Survey of multifidelity methodsin uncertainty propagation, inference, and optimization," *Society for Industrial and Applied Mathematics*, Vol. 60, No. 3, 2018, pp. 550–591.

[6] Blatman, G., "Adaptive sparse polynomial chaos expansion for uncertainty propagation and sensitivity analysis," PhD Thesis, Blaise Pascal University, France, 2009.

[7] Krige, D. G., "Statistical approach to some basic mine valuation problems on the Witwatersrand," *Journal of the Chemical, Metallurgical and Mining Engineering Society of South Africa*, Vol. 52, No. 6, 1951, pp. 119–139.

[8] Schobi, R., Sudret, B., and Wairt, J., "Polynomial-chaos-based kriging," *International Journal of Uncertainty Quantification*, Vol. 5, 2015, pp. 193–206.

[9] Bouhlel, M. A., Bartoli, N., Otsmane, A., and Morlier, J., "Improving Kriging Surrogates of High-Dimensional Design Models by Partial Least Squares Dimension Reduction," *Structural and Multidisciplinary Optimization*, Vol. 53, No. 5, 2016, p. 935–952. doi:10.1007/s00158-015-1395-9.

[10] Bouhlel, M. A., Bartoli, N., Otsmane, A., and Morlier, J., "An Improved Approach for Estimating the Hyperparameters of the Kriging Model for High-Dimensional Problems through the Partial Least Squares Method," *Mathematical Problems in Engineering*, Vol. 2016, No. 5, 2016. doi:10.1155/2016/6723410.

[11] Li, D., Wilson, P. A., and Jiong, Z., "An Improved Support Vector Regression and Its Modelling of Manoeuvring Performance in Multidisciplinary Ship Design Optimization," *International Journal of Modelling and Simulation*, Vol. 35, 2015, pp. 122–128.

[12] Forrester, A. I. J., Sobester, A., and Keane, A. J., *Engineering design via surrogate modelling: A practical guide*, John Wiley and Sons, Ltd, United Kingdom, 2008.

[13] 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*, Vol. 401, 2020, p. 109020. doi:10.1016/j.jcp.2019.109020.

[14] Raissi, M., and Karniadakis, G., "Deep Multi-fidelity Gaussian Processes," arXiv:1604.07484, 2016.

[15] Perdikaris, P., Raissi, M., Damianou, A., Lawrence, N. D., and Karniadakis, G. E., "Nonlinear information fusion algorithms for data-efficient multi-fidelity modelling," *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, Vol. 473, No. 2198, 2017, p. 20160751. doi:10.1098/rspa.2016.0751.

[16] Bouhlel, M. A., and Martins, J. R., "Gradient-Enhanced Kriging for High-Dimensional Problems," *Engineering with Computers*, Vol. 35, No. 1, 2019, p. 157–173.

[17] Bouhlel, M. A., He, S., and Martins, J., "Scalable gradient–enhanced artificial neural networks for airfoil shape design in the subsonic and transonic regimes," *Structural and Multidisciplinary Optimization*, Vol. 61, No. 4, 2020, p. 1363–1376. doi:10.1007/s00158-020-02488-5.

[18] Kennedy, M. C., and O'Hagan, A., "Predicting the output from a complex computer code when fast approximations are available," *Biometrika Trust*, Vol. 87, No. 1, 2000, pp. 1–13.

[19] Du, X., and Leifsson, L., "Multifidelity Modeling by Polynomial Chaos-Based Cokriging to Enable Efficient Model-Based Reliability Analysis of NDT Systems," *Journal of Nondestructive Evaluation*, Vol. 39, No. 1, 2020.

[20] Deng, Y., "Multifidelity Data Fusion via Gradient-Enhanced Gaussian Process Regression," *Communications in Computational Physics*, Vol. 28, No. 5, 2020, p. 1812–1837. doi:10.4208/cicp.oa-2020-0151.

[21] Goodfellow, I., Bengio, Y., and Courville, A., *Deep Learning*, The MIT Press, Cambridge, MA, 2016.

[22] Czarnecki, W. M., Osindero, S., Jaderberg, M., Świrszcz, G., and Pascanu, R., "Sobolev Training for Neural Networks," *31st Conference on Neural Information Processing Systems (NIPS 2017)*, Long Beach, CA, USA, December 4-9, 2017.

[23] Nagawkar, J., and Leifsson, L., "Gradient-Enhanced Multifidelity Neural Networks for High-Dimensional Function Approximation," *Proceedings of the ASME 2021 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference. Volume 3B: 47th Design Automation Conference (DAC)*, Virtual, Online, August 17–19, 2021. doi:10.1115/DETC2021-70502.

[24] He, X., Li, J., Mader, C. A., Yildirim, A., and Martins, J. R. R. A., "Robust aerodynamic shape optimization - from a circle to an airfoil," *Aerospace Science and Technology*, Vol. 87, 2019, pp. 48–61.

[25] Lee, C., Koo, D., Telidetzki, K., Buckley, H., Gagnon, H., and Zingg, D. W., "Aerodynamic Shape Optimization of Benchmark Problems Using Jetstream," *53rd AIAA Aerospace Sciences Meeting*, Kissimmee, FL, January 5-9, 2015. doi:10.2514/6.2015-0262.

[26] Bisson, F., Nadarajah, S., and Shi-Dong, D., "Adjoint-Based Aerodynamic Optimization Framework," *52nd Aerospace Sciences Meeting*, National Harbor, Maryland, USA, January 13-17, 2014. doi:10.2514/6.2014-0412.

[27] Anderson, G. R., Nemec, M., and Aftosmis, M. J., "Aerodynamic Shape Optimization Benchmarks with Error Control and Automatic Parameterization," *53rd AIAA Aerospace Sciences Meeting, 2015*, Kissimmee, FL, January 5-9, 2015. doi:10.2514/6.2015-1719.

[28] Zhang, Y., Han, Z.-H., Shi, L., and Song, W.-P., "Multi-round Surrogate-based Optimization for Benchmark Aerodynamic Design Problems," *54th AIAA Aerospace Sciences Meeting, 2016*, San Diego, CA, January 4-8, 2016. doi:10.2514/6.2016-1545.

[29] Iuliano, E., "Global optimization of benchmark aerodynamic cases using physics-based surrogate models," *Aerospace Science and Technology*, Vol. 67, 2017, pp. 273 – 286. doi:10.1016/j.ast.2017.04.013.

[30] Nagawkar, J., Leifsson, L., and Du, X., "Applications of Polynomial Chaos-Based Cokriging to Aerodynamic Design Optimization Benchmark Problems," *AIAA Scitech 2020 Forum*, Orlando Florida, 6-10 January, 2020.

[31] Nagawkar, J., Ren, J., Du, X., Leifsson, L., and Koziel, S., "Single- and Multipoint Aerodynamic Shape Optimization Using Multifidelity Models and Manifold Mapping," *Journal of Aircraft*, Vol. 58, No. 3, 2021, pp. 591–608. doi:10.2514/1.C035297.

[32] 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*, Vol. 21, No. 2, 1979, pp. 239–245.

[33] Kingma, D. P., and Ba, J., “Adam: A Method for Stochastic Optimization,” *ICLR*, San Diego, CA, May 7–9, 2014.

[34] Chauvin, Y., and Rumelhart, D. E., *Backpropagation: theory, architectures, and applications*, Psychology press, Hillsdale, NJ, 1995.

[35] Rall, L. B., *Automatic Differentiation: Techniques and Applications. Lecture Notes in Computer Science*, Springer-Verlag, Berlin, Germany, 1981.

[36] Sederberg, T. W., and Parry, S. R., “Free-Form Deformation of Solid Geometric Models,” *Proceedings of the 13th annual conference on Computer graphics and interactive techniques*, Dallas, August 18–22, 1986. doi:10.1145/15886.15903.

[37] Kraft, D., “A Software Package for Sequential Quadratic Programming,” *Technical Report DFVLR-FB 88-28, Institut für Dynamik der Flugsysteme, Oberpfaffenhofen, July*, 1988.

[38] Kenway, G. K., Kennedy, G. J., and Martins, J. R. R. A., “A CAD-Free Approach to High-Fidelity Aerostructural Optimization,” *Proceedings of the 13th AIAA/ISSMO Multidisciplinary Analysis Optimization Conference*, Fort Worth, TX, 2010. doi: 10.2514/6.2010-9231, AIAA 2010-9231.

[39] Mader, C. A., Kenway, G. K., Yildirim, A., and Martins, J. R., “ADflow: an open-source computational fluid dynamics solver for aerodynamic and multidisciplinary optimization,” *Journal of Aerospace Information Systems*, Vol. 17, No. 9, 2020, pp. 508–527.

[40] Spalart, P. R., and Allmaras, S. R., “A One Equation Turbulence Model for Aerodynamic Flows,” *30th AIAA Aerospace Sciences Meeting and Exhibit*, Vol. 92-0439, Reno, NV, 6-9 January 1992. doi:10.2514/6.1992-439.

[41] Yildirim, A., Kenway, G. K., Mader, C. A., and Martins, J. R., “A Jacobian-free approximate Newton–Krylov startup strategy for RANS simulations,” *Journal of Computational Physics*, Vol. 397, 2019, p. 108741. doi:10.1016/j.jcp.2019.06.018.

[42] Kenway, G. K., Mader, C. A., He, P., and Martins, J. R., “Effective adjoint approaches for computational fluid dynamics,” *Progress in Aerospace Sciences*, Vol. 110, 2019, p. 100542.

[43] Secco, N., Kenway, G. K. W., He, P., Mader, C. A., and Martins, J. R. R. A., “Efficient Mesh Generation and Deformation for Aerodynamic Shape Optimization,” *AIAA Journal*, 2021. doi:10.2514/1.J059491.