The 9th European Congress on Computational Methods in Applied Sciences and Engineering
ECCOMAS Congress 2024
3–7 June 2024, Lisboa, Portugal

MACHINE LEARNING-BASED SURROGATES FOR UNCERTAINTY QUANTIFICATION AND DESIGN UNDER UNCERTAINTIES

VARVARA G. ASOUTI[1,2], MARINA G. KONTOU[2] AND KYRIAKOS C. GIANNAKOGLOU[2]

[1] FOSS: Flow & Optimization, Software & Services, 18531 Piraeus, Greece
e-mail: vasouti@fossgp.com

[2] Parallel CFD & Optimization Unit, School of Mechanical Engineering
National Technical University of Athens, 15772 Athens, Greece
e-mail: mkontou@mail.ntua.gr, kgianna@mail.ntua.gr

Key words: Machine Learning, Uncertainty Quantification, Design Under Uncertainties

Summary. By means of three external aerodynamics cases, two isolated airfoils and one wing, this paper presents an assessment of the use of Machine Learning (ML) based models in Uncertainty Quantification (UQ) and Design under Uncertainties. Deep Neural Networks (DNNs) are used as surrogates to CFD codes to support UQ which is performed using the otherwise prohibitively expensive Monte-Carlo as well as the non-intrusive, Gauss Quadrature and regression-based, Polynomial Chaos Expansion methods. In all cases, UQ is performed and comparisons are made; in one of them, the UQ algorithm is incorporated into an aerodynamic shape optimization under uncertainties, via a gradient-free method. In the examined applications, uncertainties are associated with the constants of the $\gamma - \tilde{Re}_{\theta t}$ transition model, the surface roughness, geometric imperfections and/or flow conditions. Regarding the ML models themselves, fully-connected DNNs and the $\lambda-$DNN developed by the group of authors are used and compared. The comparison is fair since, before being used for UQ, the hyperparameters of both networks are optimized using an evolutionary algorithm.

1 INTRODUCTION

Aerodynamic Computational Fluid Dynamics (CFD)-based analyses and/or optimizations commonly consider uncertainties associated with the boundary conditions, the flow model constants, geometric imperfections, etc. To account for them, Usncertainty Quantification (UQ) of the flow system responses, a.k.a. Quantities of Interest (QoI), is necessary.

Monte Carlo (MC) methods perform an exhaustive sampling which makes them non-affordable when it comes to UQ in problems with computationally expensive tools such as a CFD s/w. More efficient approaches, such as (the non-intrusive variant of) the Polynomial Chaos Expansion (PCE), [1], are usually preferred in engineering problems. Even in this case, though, the number of simulations required by the UQ increases exponentially with the number of uncertain variables. Thus, in the literature, efficient UQ methods for aerodynamic problems with many uncertain variables, is still an open question.

On the other hand, during the last years, Machine Learning (ML) has become very popular across various engineering fields, [2, 3, 4]. It can also be used to support the UQ by deriving dependable ML-based surrogates to computationally expensive CFD tools, [5].

This paper investigates the use of Deep Neural Networks (DNNs) as a means to support the CFD s/w and supporting either the MC or the PCE-based UQ methods. This is can be seen as extension of the work presented in [6] (from the group of authors, also) that used Radial Basis Function (RBF) networks, for the same purpose. Herein, both "standard" fully connected and $\lambda-$DNN architectures are used and compared. DNNs' configurations are optimized (instead of being arbitrarily defined by the user or determined through a trial-and-error process) using an Evolutionary Algorithm (EA) for minimum prediction error. The so-trained DNNs along with the aforementioned UQ methods are used to quantify uncertainties associated with the transition model constants and/or aerodynamic flow conditions and/or geometric imperfections (modeled through the Karhunen–Loève, KL, expansion) and/or the surface roughness, for flows around isolated airfoils and wings.

2 METHODS AND TOOLS

2.1 The CFD Tool

The CFD tool used in this paper is the in-house GPU-accelerated code PUMA, [7], which solves the Reynolds-Averaged Navier-Stokes (RANS) equations for steady flows of compressible fluids. In 3D, the mean-flow equations read

$$\frac{\partial U_n}{\partial \tau} + \frac{\partial f_{nk}^{\text{inv}}}{\partial x_k} - \frac{\partial f_{nk}^{\text{vis}}}{\partial x_k} = 0 \tag{1}$$

with $n \in [1,5]$ and $k \in [1,3]$. Eqs. 1, are solved for the conservative flow variables $U = [\rho \;\; \rho v_1 \;\; \rho v_2 \;\; \rho v_3 \;\; \rho E]^T$, where ρ is the fluid's density, v_k are the Cartesian velocity components, $E = \frac{p}{\gamma-1} + \frac{1}{2}\rho v_k v_k$ the total energy per unit mass and p the pressure. τ is the pseudo-time step and x_k the Cartesian coordinates. f_{nk}^{inv} are the inviscid fluxes and f_{nk}^{vis} the viscous ones which involve the stress tensor $\tau_{km} = (\mu + \mu_t)\left(\frac{\partial v_k}{\partial x_m} + \frac{\partial v_m}{\partial x_k} - \frac{2}{3}\delta_{km}\frac{\partial v_\ell}{\partial x_\ell}\right)$. Turbulence is modeled by the one-equation Spalart-Allmaras model, [8], computing $(\tilde{\nu})$ by solving

$$\frac{\partial(\rho\tilde{\nu})}{\partial\tau} + \frac{\partial(\rho v_k\tilde{\nu})}{\partial x_k} - \frac{\rho}{\sigma}\left\{\frac{\partial}{\partial x_k}\left[(\nu+\tilde{\nu})\frac{\partial\tilde{\nu}}{\partial x_k}\right] + c_{b_2}\frac{\partial\tilde{\nu}}{\partial x_k}\frac{\partial\tilde{\nu}}{\partial x_k}\right\} - \tilde{P}_{\tilde{\nu}} + D_{\tilde{\nu}} = 0 \tag{2}$$

and turbulent viscosity μ_t is computed as $\mu_t = \rho\tilde{\nu} f_{v1}$. All other terms and constants are defined in [8]. To predict laminar-to-turbulent transition, the intermittency γ and transition momentum-thickness Reynolds number $\tilde{Re}_{\theta t}$ fields are additionally computed by solving the two PDEs of the $\gamma-\tilde{Re}_{\theta t}$ transition model, [9], which read

$$\frac{\partial(\rho\gamma)}{\partial\tau} + \frac{\partial\left(\rho v_k\gamma\right)}{\partial x_k} - \frac{\partial}{\partial x_k}\left[\left(\mu+\frac{\mu_t}{\sigma_f}\right)\frac{\partial\gamma}{\partial x_k}\right] - P_\gamma + E_\gamma = 0 \tag{3}$$

$$\frac{\partial\left(\rho\tilde{Re}_{\theta t}\right)}{\partial\tau} + \frac{\partial\left(\rho v_k\tilde{Re}_{\theta t}\right)}{\partial x_k} - \frac{\partial}{\partial x_k}\left[\sigma_{\theta,t}\left(\mu+\mu_t\right)\frac{\partial\tilde{Re}_{\theta t}}{\partial x_k}\right] - P_{\theta,t} - D_{SCF} = 0 \tag{4}$$

In the transition model equations, the production and destruction terms, are

$$\begin{aligned} P_\gamma &= \rho c_{\alpha_1} F_{length} F_{onset}\left[\phi_{-300}\left(\Omega,\Omega_{thres}\right)\right]\sqrt{\gamma}\left(1-c_{\epsilon_1}\gamma\right) \\ E_\gamma &= \rho c_{\alpha_2} F_{turb}\left[\phi_{-300}\left(\Omega,\Omega_{thres}\right)\right]\gamma\left(c_{\epsilon_2}\gamma-1\right) \end{aligned} \tag{5}$$

$$P_{\theta,t} = \rho \frac{c_{\theta,t}}{\mathrm{T}} \left(Re^{eq}_{\theta,t} - \tilde{R}e_{\theta t} \right) (1 - F_{\theta t})$$
$$D_{SCF} = c_{\theta,t} \frac{\rho}{\mathrm{T}} c_{crossflow} \phi_{-300} \Big(Re_{SCF} - \tilde{R}e_{\theta t}, 0 \Big) F_{\theta t} \tag{6}$$

where $\phi_q(\alpha, \beta)$ is a smooth min. /max. surrogate function for two variables α, β (with $q<0$ for the min. and $q>0$ for the max. function), Ω the vorticity and Ω_{thres} a limiting function based on freestream quantities. Surface roughness h_{rms} is present into the Re_{SCF} term which reads $Re_{SCF} = -35.088 \ln(h_{rms}/\theta_t) + 319.51 + f(\mathcal{D}H_{CF+}) - f(\mathcal{D}H_{CF-})$. The expressions of other terms can be found in [9], where the values of the model constants are: $c_{\alpha_1} = 2$, $c_{\alpha_2} = 0.06$, $c_{\epsilon_1} = 1$, $c_{\epsilon_2} = 50$, $c_{\theta,t} = 0.03$, $c_{crossflow} = 0.6$, $\sigma_{\theta,t} = 2$ and $\sigma_f = 1$. The γ field computed by solving Eq. 3, affects the production term of the Spalart-Allmaras model equation which becomes $\tilde{P}_{\tilde{\nu}} = \gamma \rho c_{b_1} \tilde{S} \tilde{\nu}$.

2.2 Uncertainty Quantification Methods

The MC method and two non-intrusive variants of the PCE are used to quantify the effect of uncertainties on the QoI. Let $\vec{c} \in R^M$ be the vector of uncertain variables where each c_i, $i \in [1, M]$ has its own probability density function. The MC method requires a very large number of evaluations to compute the statistical moments of any QoI, $J(\vec{c})$. On the other hand, the truncated PCE of J yields

$$J(\vec{c}) = \sum_{i=0}^{Q} J_i H_i(\vec{c}) \tag{7}$$

where $H_i(\vec{c})$ are multivariate orthogonal polynomials, selected once the probability distribution of each uncertain variable is known. The number $Q+1 = \frac{(M+K)!}{M!K!}$ of expansion terms depends on the number of uncertain variables M and the user-defined maximum degree K of the polynomials, a.k.a. chaos order. After computing the $(Q+1)$ coefficients J_i, the mean (μ_J) and standard deviation (σ_J) of J are given by, [1],

$$\mu_J = J_0 \ , \quad \sigma_J = \left(\sum_{i=1}^{Q} J_i^2 \right)^{1/2} \tag{8}$$

A way to compute J_i is by performing Galerkin projections of J to $H_i(\vec{c})$ and, then, computing the resulting integrals using the Gauss Quadrature (GQ) rules; a full-grid integration asks for $(K+1)^M$ calls to the s/w computing J. This stands for the GQ-based PCE (abbreviated to gPCE).

Alternatively, the J_i can be computed using regression (rPCE). J is computed at S value-sets of $\vec{c}$ (by sampling the uncertain space through, for instance, the Latin Hypercube Sampling method; LHS). Then, once Eq. 7 is valid for all samples, leading to the following system of S equations

$$\begin{bmatrix} H_0(\vec{c}_1) & \dots & H_Q(\vec{c}_1) \\ \vdots & \ddots & \vdots \\ H_0(\vec{c}_S) & \dots & H_Q(\vec{c}_S) \end{bmatrix} \begin{bmatrix} J_0 \\ \vdots \\ J_Q \end{bmatrix} = \begin{bmatrix} J(\vec{c}_1) \\ \vdots \\ J(\vec{c}_S) \end{bmatrix} \tag{9}$$

with $(Q+1)$ unknowns, can be written. S should exceed $Q+1$, in which case Eq. 9 stands for a least-squares problem. In this work, oversampling by 6 is used, i.e. $S \simeq 6(Q+1)$.

2.3 Machine Learning Tools

This work makes use of both the "standard" fully connected network (fc-DNN) and the two-branch $\lambda-$DNN architecture proposed in [10] by the group of authors. The latter is named after its two-branch shape which looks like the Greek letter λ, see Fig. 1. It includes two separate input branches merged into the main (output) one. Each branch first processes data through its fully-connected layers; then, the main branch further processes merged signals through its own fully-connected layers and concludes to the network output(s). The two branches allow the separate processing of input data of different nature such as geometric and flow data. Thus, different features from each branch are extracted, before merging them.

The hyperparameters of both networks are optimized for minimum prediction error. In the fc-DNN, this helps deciding the optimal combination of the numbers of layers and neurons per layer as well as the type of activation functions. Numbers of neurons are always in powers of 2 and the activation functions are selected among the Rectified Linear Unit (ReLU), Gaussian Error Linear Unit (GELU), hyperbolic tangent (*tanh*) and sigmoid. In the $\lambda-$DNN, for each one of the left/right input and the output branches, the numbers of layers and neurons per layer are similarly optimized, over and above to the activation functions. An additional variable is used/to define the (common) number of neurons at the last layer of the left and right branches in order to ensure that these can merge to form the output branch.

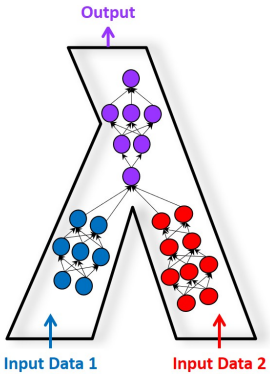

Figure 1: The $\lambda-$DNN architecture. Each layer (indicated with a few filled circles) on each branch comprises a number of neurons.

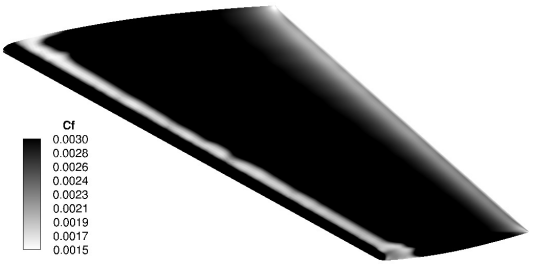

Figure 2: ONERA M6 Wing - Skin friction coefficient (C_f) over the wing suction side, as computed using the CFD tool.

2.4 Optimization Method - The Dual Role of Evolutionary Algorithms

This paper involves two different optimizations, for a different purpose each, namely to:

(i) determine the optimal DNN configuration for each QoI, and

(ii) find the best performing aerodynamic shape, by considering uncertainties.

Both optimizations are based on a (μ, λ) EA with μ parents and λ offspring, namely the NTUA in-house EASY s/w, [11]. During the search of the optimal solution, EASY uses on-line trained "personalized" surrogate models or metamodels replacing the evaluation s/w (even if this is a quite fast DNN), giving rise to the so–called Metamodel-Assisted EA (MAEA). The metamodels built in EASY are standard RBF networks and should be distinguished from the DNN-based surrogates. The RBF-based metamodels start being used after having evaluated (on the user-defined

evaluation tool) and stored (in the EA-database) at least T^{MM} (user-defined) individuals during the previous generations. Then, for each newly generated offspring, a "personalized" metamodel is trained on the "closest" individuals found in the EA-database used to approximately evaluate it; only a few potentially most promising individuals ($\lambda_e \ll \lambda$; λ_e is also user-defined) in the current offspring population are re-evaluated on the user-defined evaluation tool.

To reduce the turnaround time, during the optimization of a DNN configuration, its training is limited to a few epochs. Therefore, once the MAEA finds the best configuration, this should adequately be trained using as many epochs as necessary.

2.5 Geometric Uncertainties Modeling

Geometric uncertainties are modeled using the KL expansion technique, according to which stochastic perturbations are superimposed onto the nominal airfoil geometry, in the normal to the wall direction; the suction and pressure side of the airfoil are separately perturbed. Perturbations are generated as

$$\sum_{i=1}^{M_{mod}} \sqrt{\lambda_i}\phi_i(s)c_i \tag{10}$$

where s is the curvilinear coordinate along each airfoil side and λ_i and ϕ_i are the eigenvalues and eigenfunctions, respectively, of the selected covariance kernel (Gaussian kernel with zero mean value). For known standard deviations of c_i, both λ_i and ϕ_i are given by closed-form expressions, [12]. Herein, the number of terms M_{mod} is selected so as $\sum_{i=1}^{M_{mod}} \lambda_i \gtrsim 0.95 \sum_{i=1}^{\infty} \lambda_i$.

3 APPLICATIONS

In all cases, one Time Unit (1 TU) is the (average) cost of performing a CFD simulation (using the tool of Sec. 2.1) in this particular case, on the generated computational grid. It is also assumed that CFD simulations on modified geometries with the same grid size have the same cost. It is obvious that TU is different in each presented case.

3.1 The ONERA M6 Wing Case

The first case deals with the transitional flow around the ONERA M6 wing, [13] at $M_\infty = 0.262$, $Re_c = 3.5 \cdot 10^6$, zero angle of attack and yaw angle and turbulence intensity $Tu = 0.2\%$. Surface roughness is $h_{rms} = 5\mu m$. The $\gamma - \tilde{Re}_{\theta t}$ transition model is used. The skin friction coefficient (C_f) field on the wing suction side computed using the CFD tool is illustrated in Fig. 2. White areas indicate the part of the wing surface along which the flow is laminar.

The transition model includes many constants calibrated against experimental data on different cases that the one studie here. So, these model constants can be non-perfectly suited to this particular problem. Additionally, measurements of surface roughness (h_{rms}) are usually characterized by uncertainties. Having these in mind, this case is used to quantify uncertainties in the drag coefficient (C_D) caused by changes in the values of four $\gamma - \tilde{Re}_{\theta t}$ model constants (c_{α_1}, c_{α_2}, c_{ϵ_2}, $c_{\theta,t}$) and the surface roughness (h_{rms}) in Eqs. 5-6. This makes $M = 5$ uncertain variables in total, which are all assumed to follow normal distributions as:

$$c_{\alpha_1} \sim \mathcal{N}(2.0, 0.2), \quad c_{\alpha_2} \sim \mathcal{N}(0.06, 0.006)$$
$$c_{\epsilon_2} \sim \mathcal{N}(50.0, 5.0), \quad c_{\theta,t} \sim \mathcal{N}(0.03, 0.003), \quad h_{rms} \sim \mathcal{N}(5 \cdot 10^{-6}, 1.6 \cdot 10^{-6})\mu m$$

Performing UQ using the gPCE on the CFD tool with $K = 2$ requires 243 ($= 3^5$) CFD runs (at a cost of 243 TUs), for the 243 Gauss nodes determined by the GQ rules, see Sec. 2.2. On the other hand, 120 samples in the uncertain variables' space are generated using the LHS method and used to train the DNNs predicting the QoI. Both fc-DNNs and $\lambda-$DNNs are trained and compared. For the $\lambda-$DNN, the transition model constants and the surface roughness are presented to the left and right input branches, respectively.

As described in Sec. 2.3, both DNN configurations result from a MAEA-based optimization using the tool described in Sec. 2.4. DNNs are separately configured for the 120 samples from the LHS and the 243 Gauss nodes, as training data; in total, two $\lambda-$DNNs and two fc-DNNs are optimized. The MAEA-based optimization of any DNN takes no more than 1 TU, including its final training. Table 1 summarizes the optimized DNN architectures.

Table 1: ONERA M6 Wing - Optimized configurations of the fc-DNN and $\lambda-$DNN architectures using the 120 samples from the LHS (top) and the 243 Gauss nodes (bottom) as training data.

	Layers	**Neurons/Layer**	**Act. Functions**
Trained on 120 samples			
$\lambda-$DNN			
Left input branch (c_{α_1}, c_{α_2}, c_{ϵ_2}, $c_{\theta,t}$)	5	$1024, 64, 32, 128, 4096$	ReLU
Right input branch (h_{rms})	3	$128, 256, 4096$	ReLU
Output branch	1	64	sigmoid
fc-DNN (c_{α_1}, c_{α_2}, c_{ϵ_2}, $c_{\theta,t}$, h_{rms})	3	$128, 2048, 32$	ReLU/*tanh*
Trained on 243 Gauss nodes			
$\lambda-$DNN			
Left input branch (c_{α_1}, c_{α_2}, c_{ϵ_2}, $c_{\theta,t}$)	2	$128, 64$	ReLU
Right input branch (h_{rms})	4	$1024, 2048, 128, 64$	ReLU
Output branch	3	$64, 64, 32$	*tanh*
fc-DNN (c_{α_1}, c_{α_2}, c_{ϵ_2}, $c_{\theta,t}$, h_{rms})	5	$256, 4096, 32, 256, 32$	ReLU/sigmoid

Error metrics, such as the percentage Mean Absolute Error (MAE %) and the Root Mean Square Error (RMSE), of DNNs trained on data of different size, are compared in Fig. 3. Overall, both networks perform very well due to their optimized architectures. Using 120 training data, the $\lambda-$DNN outperforms the fc-DNN. The fc-DNN reaches the accuracy of the $\lambda-$DNN by increasing the number of training data to 243.

Figure 3: ONERA M6 Wing - MAE % (left) and RMSE (right) error metrics of $\lambda-$DNNs and fc-DNNs.

The so-trained DNNs are then used to support the UQ using gPCE and MC methods. Results are, also, compared with those of the gPCE and rPCE supported by the CFD tool (denoted as gPCE_{CFD} and rPCE_{CFD} respectively; hereafter, the name of each method is indexed by the evaluation tool used to support it), and are summarized in Table 2. Column labeled as "Samples" indicates the number of data processed for UQ, while column "TUs" stands for the computational cost to perform the CFD evaluations and train the DNNs (in case these are used). The outcome of gPCE_{CFD} (using the 243 Gauss nodes) and rPCE_{CFD} (with the 120 samples) is presented in Table 2, top. The statistical moments computed by the rPCE_{CFD} are in good agreement with those from gPCE_{CFD}, considered to yield reference results. The computed mean values are practically identical with small differences ($\sim 2.3\%$) in standard deviation.

As an extra test, DNNs trained on the 120 samples are also used to predict the QoI values at the 243 Gauss nodes (ignoring, on purpose, the available CFD results) and, then, perform a gPCE UQ. The outcome of this try is given in Table 2, center. The mean value computed using either network is in very good agreement with the gPCE_{CFD}. Regarding the standard deviation, the use of the $\lambda-$DNN yields by far better results, i.e. with error less than 0.1% compared to the $\sim 1\%$ error of the fc-DNN. This reconfirms the very good prediction accuracy of the $\lambda-$DNN when trained on 120 samples.

Finally, the outcome of the MC method implementing DNNs trained on both the 120 samples and the 243 Gauss nodes is also given in Table 2, bottom. In any MC run, 5×10^6 replicates are generated and predicted. The $\text{MC}_{\lambda-\text{DNN}}$ and $\text{MC}_{\text{fc-DNN}}$ yield practically the same μ_{C_D}; the percentage error, compared to the outcome of gPCE_{CFD}, is less than 0.15% when trained on 120 samples and becomes even smaller by increasing the number of training data. The σ_{C_D} values computed by $\text{MC}_{\lambda-\text{DNN}}$ are quite closer to those of the gPCE_{CFD}, irrespective of the number of training data. For the $\text{MC}_{\text{fc-DNN}}$ to reach the same level of accuracy, more training patterns are required.

To further explore the potential of the $\lambda-$DNN, thiss was additionally trained using only 60 samples (randomly selected among the 120 ones generated using the LHS) and used to support MC. The outcome of this study is also included in Table 2. Overall, it is shown that a $\lambda-$DNN may to support the MC method with acceptable accuracy (compared to the gPCE_{CFD}) even when trained with less data compared to the fc-DNN. These results are achieved with up to $\sim 75\%$ reduction in the computational cost.

3.2 The NACA16-103 Airfoil Case

The second case is dealing with the flow around a NACA16-103 profile airfoil which represents an open rotor blade design, provided by SAFRAN, in the framework of the NEXTAIR project (see acknowledgement). The flow conditions correspond to cruise at an altitude of 37000ft; $a_\infty = 1^o$, $u_\infty = 252.28$m/s, $Re_c = 3.6 \cdot 10^6$ and static temperature $T_\infty = 216.65$K. At these conditions, a shock wave appears half-chord on the suction side, Fig. 4.

Both operational and geometric uncertainties are considered. The far-field temperature and velocity vary stochastically; the differences δu_∞ and δT_∞ from reference values u_∞ and T_∞ follow a Beta distribution as:

$$\delta u_\infty \sim \text{Beta}(4.0, 4.0, -1.0, 1.0), \quad \delta T_\infty \sim \text{Beta}(4.0, 4.0, -2.0, 2.0)$$

where the first two terms in the parentheses are the beta distribution parameters (α, β) whereas the last two the bounds of the distribution. Next to them, geometric uncertainties are modeled

Table 2: ONERA M6 Wing - Statistical moments of the C_D. Comparison of UQ methods supported either by the CFD or the DNNs trained on the 234 Gauss nodes, the 120 and 60 samples.

UQ method$_{\text{Tool}}$	**Samples**	**TUs**	μ_{C_D}	σ_{C_D}
gPCE_{CFD}	243	243	0.0079933	0.00026906
rPCE_{CFD}	120	120	0.0079928	0.00027554
$\text{gPCE}_{\lambda-\text{DNN}}$	243	121	0.0080045	0.00026882
$\text{gPCE}_{\text{fc-DNN}}$	243	121	0.0080060	0.00026629
$\text{MC}_{\lambda-\text{DNN}}$	5×10^6	244	0.0079860	0.00026862
$\text{MC}_{\text{fc-DNN}}$	5×10^6	244	0.0079927	0.00027036
$\text{MC}_{\lambda-\text{DNN}}$	5×10^6	121	0.0080047	0.00026815
$\text{MC}_{\text{fc-DNN}}$	5×10^6	121	0.0080058	0.00026293
$\text{MC}_{\lambda-\text{DNN}}$	5×10^6	61	0.0080005	0.00026784

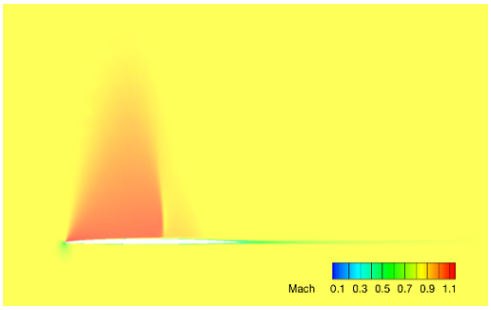

Figure 4: NACA16-103 Airfoil - Mach number field computed by the CFD tool.

as is Sec. 2.5; 10 uncertain variables (5 per airfoil side) following a normal distribution with zero mean value and unit standard deviation (i.e. $\mathcal{N}(0.0, 1.0)$) are considered. Thus, the total number of uncertain variables is $M = 12$. The QoIs are the lift (C_L) and drag (C_D) coefficients.

Initially, 300 samples in the c_i space were generated using LHS and simulated on the CFD tool. This set was further enriched (the criterion being the reduction of the variance in the drag prediction by a surrogate developed by SAFRAN) to 600 samples, evaluated on the CFD tool. Using the 600 samples, the fc-DNN configuration was optimized using MAEA. For the $\lambda-$DNN, the 300 initial samples proved to be enough and there was no need to resort to the 600 training patterns. The $\lambda-$DNN architecture utilizes the 10 geometric uncertainties as inputs to the left whereas the two flow conditions are presented to the right branch. The optimized configurations are given in Table 3. The cost for finding and training the best DNN configuration is no more than 2 TUs. The trained DNNs are used to evaluate 50×10^6 random samples, to be used by the MC method. Given the huge number of MC samples, the I/O and prediction cost is not negligible at all; for $M = 12$ uncertain variables and 50×10^6 samples this amounts to 1.25 TUs. Thus, the overall cost for training and using any of the two networks to support MC is 4.5 TUs which should be added either to the 300 or 600 TUs spent for evaluating the samples used to train the $\lambda-$DNN and the fc-DNN, respectively.

Results from the $\text{MC}_{\lambda-\text{DNN}}$ and $\text{MC}_{\text{fc-DNN}}$ are summarised in Table 4 and compared to

the outcome of the rPCE_{CFD} (with both 300 and 600 samples). Given that, for chaos order $K = 2$, $(Q + 1) = 91$ J_i coefficients must be computed, the use of either 300 or 600 samples ensures an oversampling ratio more than 3 and ~ 7, respectively. The two rPCE_{CFD} UQ runs lead to small differences in the statistical moments of C_D. A working hypothesis is made that the rPCE_{CFD}, using 600 samples, computes reference statistical moments. It can be seen that the $\text{MC}_{\lambda-\text{DNN}}$ yields more accurate predictions than the $\text{MC}_{\text{fc-DNN}}$, especially for σ_{C_D}. The $\text{MC}_{\lambda-\text{DNN}}$ predictions are also in very good agreement with the rPCE_{CFD} results on the 600 samples. Overall, the $\text{MC}_{\lambda-\text{DNN}}$, compared either with the $\text{MC}_{\text{fc-DNN}}$ or the rPCE_{CFD}, provides accurate predictions at half of the computational cost.

Table 3: NACA16-103 Airfoil - Configuration of the fc-DNN and $\lambda-$DNN architectures predicting the two QoIs (C_D and C_L).

	Layers	**Neurons/Layer**	**Act. Functions**
$\lambda-$DNN for C_D			
Left input branch (10 geometric vars)	3	$4096, 512, 512$	ReLU
Right input branch (2 flow conditions)	5	$4096, 32, 256, 32, 512$	ReLU
Output branch	2	$1024, 4096$	GELU
$\lambda-$DNN for C_L			
Left input branch (10 geometric vars)	2	$2048, 64$	ReLU
Right input branch (2 flow conditions)	2	$2048, 64$	ReLU
Output Branch	3	$512, 256, 64$	sigmoid
fc-DNN for C_D	7	$2048, 64, 2048, 512, 1024, 128, 512$	GELU/*tanh*
fc-DNN for C_L	4	$64, 32, 64, 64$	ReLU/GELU

Table 4: NACA16-103 Airfoil - Statistical moments of the QoIs. Comparison of different UQ methods supported either by the CFD or the fc-DNN and $\lambda-$DNN, trained on 600 and 300 samples, respectively.

UQ method$_{\text{Tool}}$	**Samples**	**TUs**	μ_{C_D}	σ_{C_D}	μ_{C_L}	σ_{C_L}
rPCE_{CFD}	600	600	0.00839345	0.00021952	0.412822	0.012137
rPCE_{CFD}	300	300	0.00837136	0.00021607	0.412770	0.012146
$\text{MC}_{\lambda-\text{DNN}}$	50×10^6	304.5	0.00839195	0.00021930	0.411961	0.011923
$\text{MC}_{\text{fc-DNN}}$	50×10^6	604.5	0.00839201	0.00023098	0.412425	0.012242

3.3 Optimization under Uncertainties of the S8052 Airfoil Case

The flow around the S8052 isolated airfoil at flow conditions: $M_\infty = 0.1$, $\alpha_\infty = 3^o$, $Re_c = 5.05 \cdot 10^6$ and $Tu = 1.28\%$ is studied. For the simulations, the nominal values of the $\gamma - \tilde{Re}_{\theta t}$ transition model are used, see Sec. 2.1.

Herein, the airfoil shape is optimized by considering geometric uncertainties. For the optimization, the unperturbed airfoil shape is parameterized using a 13×9 NURBS control lattice, as in Fig. 5. The 15 red control points can be displaced in the normal-to-the-chord direction, giving rise to $N = 15$ design variables, in total. Geometric uncertainties are modeled as in Sec. 2.5; $M = 18$ uncertain variables (9 per airfoil side) following a normal distribution ($\mathcal{N}(0.0, 0.3)$) are considered. The drag-to-lift ratio is selected as the QoI, i.e. $J = C_D/C_L$ and the objective function to be minimized during the shape optimization is $F = \mu_J + \sigma_J$.

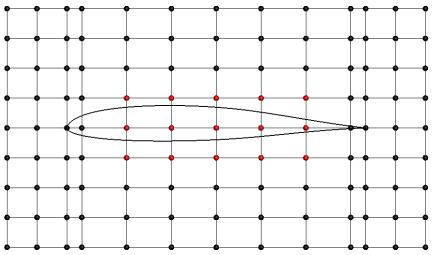

Figure 5: S8052 Airfoil - NURBS control lattice used to parameterize the airfoil shape. Only control points marked in red can be displaced.

Since, in the previous cases, the $\lambda-$DNN provided more accurate predictions even when trained on less data, only the $\lambda-$DNN architecture is used below. In specific, two $\lambda-$DNNs are trained to predict the C_D and C_L values; their ratio (J) as well as its statistical moments are, then, computed. To train the networks, the design space is sampled (based on the ranges of control points in red, Fig. 5) using the LHS method; 200 different airfoil geometries are generated and evaluated on the CFD tool.

The design variables vector and the 198 airfoil nodal coordinates (of the unperturbed shape) are used as inputs to the two branches of the $\lambda-$DNN. This selection allows for the network to accurately predict C_L and C_D, not only in the case of a new airfoil geometry resulting during the optimization for different value-sets of the design variables, but also in case of geometric imperfections (the effect of which is much smaller). The cost for evaluating the training patterns is 200 TUs. The additional cost for finding the optimal $\lambda-$DNN architecture and training the network for both C_L and C_D is rounded up to 12 TUs.

The $\lambda-$DNN is used to support both the MC method and rPCE for the computation of μ_J and σ_J. For 18 uncertain variables, the UQ for each candidate airfoil geometry with the MC method would require the generation of, at least, 10^6 perturbed geometries; in this case, the computational cost of UQ is $\sim$1 TU. On the other hand, generating 1000 perturbed geometries is adequate to support the rPCE (for chaos order 2, in which case 190 coefficients must be computed) and the overall computational cost of the UQ is 0.125 TUs. During the shape optimization run, the $\text{rPCE}_{\lambda-\text{DNN}}$ is used as the evaluation tool. The latter predicts J for the baseline geometry with $\sim 2.5\%$ error.

A termination criterion of 400 TUs is set for the shape optimization run. Given that the 212 TUs were spent for evaluating the samples and setting up the optimal $\lambda-$DNN, the shape optimization should not exceed 1500 evaluations ($1500 \times 0.125 \simeq 188$ TUs to reach the termination criterion). The optimization is carried out by means of a $(10, 30)$ MAEA. The use of the RBF metamodels in the MAEA starts after having evaluated (on the $\text{rPCE}_{\lambda-\text{DNN}}$) $T^{MM}=300$ airfoils and only $\lambda_e=5$ members in each generation are re-evaluated on the $\text{rPCE}_{\lambda-\text{DNN}}$. The optimization convergence history is shown in Fig. 6.

The optimized shape is re-evaluated on the CFD tool and the objective function value as well as the statistical moments of the QoI are given in Table 5. It can be seen that the optimized geometry is better than the baseline one, with $\sim$17% reduction in F which is mainly attributed to the reduction in μ_J.

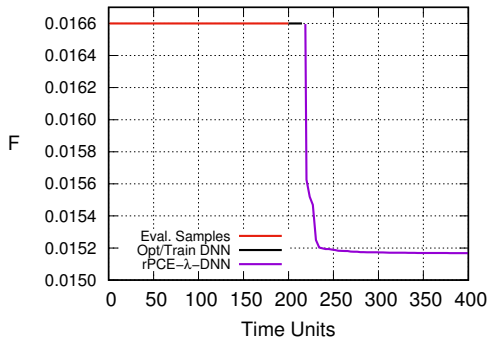

Figure 6: S8052 Airfoil - Convergence history (purple line) of the shape optimization run under geometric uncertainties; the vertical axis corresponds to values of F predicted by the $\text{rPCE}_{\lambda-\text{DNN}}$ model. The red line (first part) corresponds to the cost of evaluating the samples while the (small) black line (in the middle) to the optimization of the $\lambda-$DNN configuration including network training.

Table 5: Comparison of the objective function $F = \mu_J + \sigma_J$ and statistical moments of $J = C_D/C_L$ of the baseline and the optimized geometries; the latter after being (re)-evaluated on the CFD tool.

Geometry	F	μ_J	σ_J
Baseline	0.0172095	0.01694	0.0002695
Optimized	0.0143655	0.01391	0.0004555

4 CONCLUSIONS

This paper contributes to the assessment of the use of ML-based tools, as surrogates to CFD analysis codes, in UQ and aerodynamic shape optimization under uncertainties. In specific, standard fully connected and $\lambda-$DNN, are used to predict the QoIs in the framework of the Monte-Carlo and regression PCE UQ methods. The two UQ studies related to the flows around a wing and the NACA16-103 airfoil, demonstrated that the DNNs may efficiently support the MC method as they reduce the overall computational cost by even 75%, compared to the same UQ method relying exclusively on the CFD tool. It was also shown that the error in predicting the first statistical moments of the lift and drag coefficients is smaller if an appropriately optimized $\lambda-$DNN architecture, instead of standard fc-DNN, is used. For a fair comparison, an MAEA was used to find the optimal configuration of both the $\lambda-$DNN and the fc-DNN. Then, an optimized $\lambda-$DNN was used to support the regression PCE UQ in CFD-based optimization under geometric uncertainties. DNNs' optimization was based on a metamodel-assisted evolutionary algorithm. Since, for each candidate solution (each new geometry generated during the evolution), hundreds of perturbed geometries must be evaluated, the QoI was computed using the $\lambda-$DNN and rPCE, at a much lower cost. This is faster than the MC with $\lambda-$DNN by eight times and faster than the use of the CFD tool by orders of magnitude.

ACKNOWLEDGMENTS

The development of ML methods for UQ are part of the NEXTAIR project which is funded by the European Union (GA number 101056732). Views and opinions expressed are however

those of the author(s) only and do not necessarily reflect those of the European Union or REA. Neither the European Union nor the REA can be held responsible for them.

REFERENCES

[1] D. Xiu and G. Karniadakis. Modeling uncertainty in flow simulations via generalized polynomial chaos. *J. Comput. Phys.*, 187(1):137–167, 2003.

[2] M. Kontou, V. Asouti, and K. Giannakoglou. DNN surrogates for turbulence closure in CFD-based shape optimization. *Appl. Soft Comput.*, 134:110013, 2023.

[3] C. Mao and Y. Jin. Uncertainty quantification study of the physics-informed machine learning models for critical heat flux prediction. *Prog. Nuclear Energy*, 170:105097, 2024.

[4] A. Psaros, X. Meng, Z. Zou, L. Guo, and G. Karniadakis. Uncertainty quantification in scientific machine learning: Methods, metrics, and comparisons. *J. Comput. Phys.*, 477:111902, 2023.

[5] Y. Liu, D. Wang, X. Sun, Y. Liu, N. Dinh, and R. Hu. Uncertainty quantification for multiphase-cfd simulations of bubbly flows: a machine learning-based bayesian approach supported by high-resolution experiments. *Reliab. Eng. Syst. Saf.*, 212:107636, 2021.

[6] V. Asouti, M. Kontou, and K. Giannakoglou. Radial basis function surrogates for uncertainty quantification and aerodynamic shape optimization under uncertainties. *Fluids*, 8(11):292, 2023.

[7] I. Kampolis, X. Trompoukis, V. Asouti, and K. Giannakoglou. CFD-based analysis and two-level aerodynamic optimization on graphics processing units. *Comput. Methods Appl. Mech. Eng.*, 199(9-12):712–722, 2010.

[8] P. Spalart and S. Allmaras. A one-equation turbulence model for aerodynamic flows. *Recherche Aerospatiale*, 1:5–21, 1994.

[9] M. Piotrowski and D. Zingg. Smooth local correlation-based transition model for the Spalart-Allmaras turbulence model. *AIAA J.*, 59(2):474–492, 2021.

[10] M. Kontou, D. Kapsoulis, I. Baklagis, X. Trompoukis, and K. Giannakoglou. λ-DNNs and their implementation in conjugate heat transfer shape optimization. *Neural Comput. Appl.*, 43:843–854, 2022.

[11] M. Karakasis and K. Giannakoglou. On the use of metamodel-assisted, multi-objective evolutionary algorithms. *Eng. Optim.*, 38(8):941–957, 2006.

[12] R.G. Ghanem and P.D Spanos. *Stochastic Finite Elements: A Spectral Approach*. Springer-Verlag, New York, 1991.

[13] V. Schmitt and J. Cousteix. Etude de la couche limite tridimensionnelle sur une aile en flèche. Technical Report No 14/1713 AN, ONERA, 1975.