Data-driven augmentation of a RANS turbulence model for Transonic Flow Prediction

ABSTRACT

Purpose The paper aims to improve Reynolds-Averaged Navier Stokes (RANS) turbulence models using a data-driven approach based on machine learning. A special focus is put on determining the optimal input features used for the machine learning model.
Methodology The Field Inversion and Machine Learning (FIML) approach is applied to the negative Spalart-Allmaras turbulence model for transonic flows over an airfoil where shock-induced separation occurs. Feature selection methods are applied to the results of the field inversion to determine the optimal input features for the machine learning model.
Findings Optimal input features and a machine learning model are developed which improve the existing negative Spalart-Allmaras turbulence model with respect to shock-induced flow separation.
Originality A comprehensive workflow is demonstrated that yields insights on which input features and which machine learning model should be used in the context of the FIML approach

Keywords RANS, data-driven turbulence modeling, machine learning, feature selection, flow separation, transonic flows
Paper type Research paper

ABBREVIATIONS

CFD	Computational Fluid Dynamics
DNS	Direct Numerical Simulation
FIML	Field Inversion and Machine Learning
LES	Large Eddy Simulation
ML	Machine Learning
pETW	pilot facility European Transonic Windtunnel
RANS	Reynolds Averaged Navier Stokes
SA-neg	Spalart-Allmaras negative turbulence model
SFS	Sequential Feature Selection

1. INTRODUCTION

Turbulent flows are fully characterized by the Navier-Stokes equations. Simulating turbulent flows hence requires solving these equations. Direct Numerical Simulations (DNS) resolve all turbulent scales and are exact, however, at the expense of enormous computational cost (Probst *et al.*, 2020). Hence, they are unfeasible for daily simulation tasks. Time-averaging the Navier-Stokes equations yields the RANS equations, which can be solved with todays computational power as no small scales must be resolved. Time-averaging however introduces a new, unknown term, which includes the Reynolds stresses and represents the impact of the turbulent fluctuations on the mean flow.

The Reynolds stresses must be modeled using a turbulence model. A common approach is to introduce the Boussinesq hypothesis, which relates the Reynolds stresses to the strainrate via a scalar quantity, the eddy viscosity. For the eddy viscosity, a multitude of algebraic, 1-equation and 2-equation models exist. Common to RANS models is that they deliver precise predictions at design conditions, but their reliability greatly deteriorates towards flow conditions at the border of the flight envelope. For example, flow separation is often not captured correctly leading to severely false predictions of lift coefficients close to maximum lift.

In recent years and with maturing machine learning methods, data-driven approaches to improve existing turbulence models have gained interest (Beck and Kurz, 2021; Duraisamy, 2021; Schmelzer *et al.*, 2020; Weatheritt and Sandberg, 2016). Here, readily available high-fidelity reference data stemming from DNS, Large Eddy Simulations (LES) or wind tunnel measurements are used to train a machine learning model to improve a given turbulence model. The present study uses the Field Inversion and Machine Learning (FIML) (Singh and Duraisamy, 2016) approach, which gained popularity in the field (Ferrero *et al.*, 2020; Holland *et al.*, 2019; Jäckel, 2022). The FIML approach introduces a correction term as a multiplier to the production term of the turbulence model. Via inverse modeling, the ideal form of the correction term is determined, and a machine learning model is trained to approximate this ideal form.

We apply FIML to the one-equation negative Spalart-Allmaras turbulence model (SA-neg) (Allmaras *et al.*, 2012). In particular, we use a database of wind tunnel measurements for the transsonic RAE2822 (Cook *et al.*, 1979) airfoil and aim to improve SA-neg for shock-induced separation. A focus of this work is the engi-

neering and selection of flow features used as inputs for the machine learning model, building a feature selection pipeline that can easily be adapted to new features and different flow phenomena. The aim of the work is to consolidate work done by among others (Holland *et al.*, 2019; Ling *et al.*, 2016; Wu *et al.*, 2018), focusing on transonic flight and shock-induced separation.

2. METHODOLOGY

2.1 Field Inversion and Machine Learning

The first generation FIML ansatz (Singh and Duraisamy, 2016) is used, termed FIML Classic in later publications (Holland *et al.*, 2019). In the first step, the field inversion, the optimal values for the correction term β are determined using inverse modeling. Here, the spatially varying β is multiplied to the turbulence production term P of the SA-neg turbulence model, see Eq. (1). In the second step, a machine learning model is trained to approximate the values of the correction term depending on input features η derived from the local flow state $\vec{U}$.

$$\frac{D\tilde{\nu}}{Dt} = \beta(\vec{U})P(\vec{U}) - D(\vec{U}) + T(\vec{U}) \tag{1}$$

Here, $\tilde{\nu}$ denotes the Spalart-Allmaras transport variable, $\vec{U}$ the local flow state, P, D, and T the turbulent production, destruction, and diffusion terms respectively. Further details of the terms of the SA-neg turbulence model are given in the original publication.

In the first step, we determine the optimal distribution of β by solving the approximate inverse problem

$$q^{\mathrm{RANS}}(\beta) \approx q^{\mathrm{ref}} \tag{2}$$

where q^{RANS} is a quantity of interest of the RANS solution that we aim to fit to the corresponding quantity of interest q^{ref} of a high-fidelity reference solution by optimizing β. Due to the non-linear nature of the RANS equations contained in Eq. (2), we cannot solve this equation directly, but we pose the problem as an optimization problem instead. For the selected quantity of interest, the pressure coefficient c_p, the associated cost function is

$$\mathscr{J} = \frac{1}{2}\sum_{i}^{N} V_i (c_{p,i}^{\mathrm{ref}} - c_{p,i}^{\mathrm{RANS}}(\beta))^2 + \lambda \frac{1}{2}\sum_{i}^{M} (\beta_i - \beta_0)^2 \tag{3}$$

The first term of the cost function is the mean squared error of the deviation in c_p between the reference data and the RANS solution, weighted by the cell volumes V_i and computed across the N cells in which reference data is available. The second term is a Tikhonov regularization which penalizes deviations from the baseline turbulence model, that is, deviations of β_i, the value of β in cell i from the default value of β, $\beta_0 = 1$, for which the turbulence model remains unchanged. This is computed across all M cells of the computational domain.

The magnitude of the regularization is adjusted via the regularization parameter λ. The regularization is introduced for two reasons. First, because the problem is ill-posed, as the number of degrees of freedom, i.e. the number of cells as β takes on a different value β_i in each cell, is usually much larger than the number of points where reference data is given. For example, the pressure coefficient distribution c_p is only available on the airfoil surface while the correction term is available throughout the flow field. Second, it allows uncertainties in the reference data to be considered, as the regularization restricts the magnitude of the turbulence model modification which prevents overfitting in case of untrustworthy reference data.

For the minimization of Eq. (3) with respect to β, a steepest descent optimizer (Nocedal and Wright, 2006) is employed, which updates β according to

$$\beta_{i+1} = \beta_i - \varepsilon \frac{dI}{d\beta}_i \tag{4}$$

Here, ε is the step size and $\frac{dI}{d\beta}$ is the gradient of cost function Eq. (3) with respect to β. This gradient is computed using the adjoint method (Dwight and Brezillon, 2006; Giles and Pierce, 2000). The step size ε is determined using the Armijo condition (Nocedal and Wright, 2006) in each optimization step. The optimization is stopped when the step size falls below a threshold ε_0.

Having obtained the optimal values for β, the next step is to generalize β to different flow geometries and flow conditions. As of now, β is available only as a function of the spatial coordinates $\vec{x}$, so it can not be easily transferred and applied to different geometries and different flow conditions. Therefore, the next step is to identify a function f_β which represents β depending on local flow features $\eta_i(\vec{U})$ instead, that is

$$f_\beta : \eta_0, \ldots, \eta_n \rightarrow \beta \tag{5}$$

The features are required to be non-dimensional, ideally they are fully local and galilean invariant as well. The selection of features considered in this study is presented in Section 4.

The function f_β is approximated using machine learning, neural networks in particular. Besides neural networks, which are by far the most often applied ML-method in the FIML framework, also conventional and multiscale Gaussian processes (Zhang and Duraisamy, 2015) and Adaptive boosting (Singh, Matai, *et al.*, 2017)

were investigated and applied. Neural networks are responsible for ongoing successes in machine learning, especially in areas such as image detection, language processing or autonomous driving (LeCun *et al.*, 2015). As regressors, they are capable of approximating any limited, continuous function arbitrarily exact with a finite number of neurons (*Universal Approximation Theorem*), making them excellent candidates for our purposes. Due to limited space, we refer the reader for details of the inner workings of neural networks to dedicated publications such as (LeCun *et al.*, 2015).

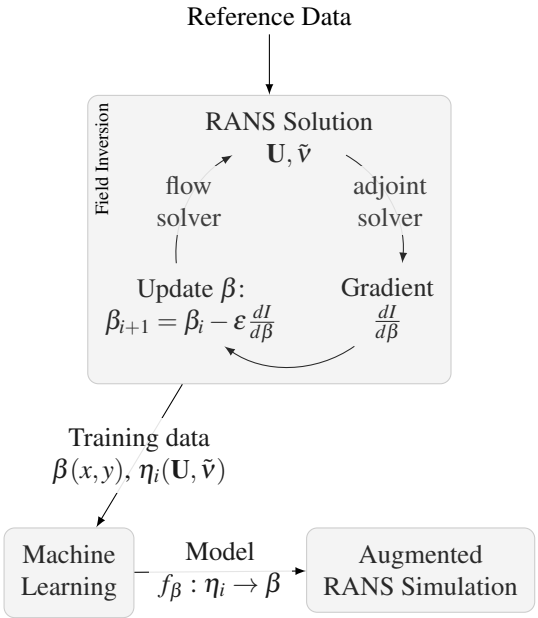

Figure 1: Workflow of FIML Classic

Figure 1 recapitulates the workflow of the FIML Classic approach.

2.2 Feature Engineering and Selection

As it might be unclear which input features are relevant for the prediction of β, we need to employ feature engineering and feature selection. Feature engineering in the present case means to derive features from the flow state which are dimensionless and ideally locally available and Galilean invariant. The need for dimensionless features stems from the fact that neural networks can not take care of dimensional consistency and that the correction term β is dimensionless itself. The desire for features being local is to enable easy availability of the features in general CFD solvers.

Many of the features listed in Section 4 were found in a literature survey, with few added due to physical considerations of the present phenomenon, shock-induced separation. After building a database of possible features, a subset of promising features was selected using the following techniques.

2.2.1 Feature Correlation

The first technique is feature correlation. Here, a correlation matrix is built which measures the correlation among the features and the features and the correction term. The used metric is *Spearman's rank correlation coefficient* r_s ("Spearman Rank Correlation Coefficient" 2008), see Eq. (6).

$$r_S = \frac{\mathrm{cov}(R(x), R(y))}{\sigma_{R(x)} \sigma_{R(y)}} \tag{6}$$

Here, $\mathrm{cov}(x,y)$ is the covariance of two variables x, y and σ_x the standard deviation of variable x. As opposed to *Pearson's correlation coefficient*, it correlates the ranks of the observations $R(x), R(y)$ instead of the observations x, y themselves. While Pearson's correlation coefficient assumes that the variables are normally distributed and detects only linear relationships, Spearman's rank correlation coefficient is able to also detect any non-linear, monotonous relationships, and makes no assumptions about the variable distribution. If the relationship between the variables is perfectly monotonous, r_S goes towards $r_S = 1$, or towards $r_S = -1$, if the relationship is inverse. In case of no discernible relationship between the variables, r_S tends towards $r_S = 0$.

Computing the correlation matrix between the possible features themselves and the features and the correction term β allows firstly, to remove redundant features, that is features that are monotonously depending on each other and secondly, to remove features that show no promising relationship with β.

2.2.2 Sequential Feature Selection

With the reduced feature set obtained, *Sequential Feature Selection* is applied. Sequential Feature Selection is not agnostic to the selected machine learning model, as it involves the ML model directly. The basic idea is to train the ML model, starting from a subset of features and then to sequentially add or remove features depending on their importance. In *Sequential Forward Selection*, the process is

1. Randomly pick a subset of k features from the whole set of features.
2. Train the ML model on this feature subset.
3. Randomly add another feature from the remaining features.
4. Retrain the ML model and keep the feature if the loss function decreases, discard it otherwise.

5. Continue with step 3 until the loss function doesn't decrease further or the size of the feature subset becomes larger than intended.

In *Sequential Backward Selection*, one starts with the full set of features and removes features until the loss function increases. *Bidirectional* SFS methods include additional inclusion/exclusion steps, which allow to include (exclude) a feature again that was previously dismissed (included).

There are three major points to be made about sequential feature selection:

- Since the algorithm is not agnostic towards the chosen ML model, it can yield substantially better results for this particular model, but the results are not universal and potentially worse for other models.
- Especially in the case of neural networks, a disadvantage is that one needs to pre-define the network hyperparameters, as either a too simple or a too complex architecture might be chosen. In this work, we use a set of neural network hyperparameters proven to work before (Jäckel, 2021), but will do a hyperparameter optimization subsequently.
- Although SFS is a greedy algorithm that doesn't necessitate an exhaustive search of all possible subsets of features, it can still become computationally expensive as for each investigated subset of features, the ML model has to be trained. Additionally, the algorithm should be run multiple times to eliminate effects due to the random initialization of the neural network parameters. In this work, computational effort was still only on the order of few minutes, using a current mid-class graphics card for NN training.

2.2.3 Engineering considerations

The previous steps leave us a shortlist of input features for the machine learning part. The final step is to train different neural networks on subsets of features from this shortlist, and apply them in the full CFD loop. While a ML model might train flawlessly, it is not ensured that its inclusion in the turbulence model and the full CFD loop will work as well. A frequent observation is that including the trained model will impact convergence heavily due to a unsmooth prediction of β. Hence, the final step is to evaluate the ML models and appropriate feature subsets according to their performance in the CFD loop.

2.3 Software

The Field Inversion and Machine Learning approach has been implemented in DLR's software ecosystem before (Jäckel, 2021), using the unstructured high-performance CFD code TAU (Schwamborn *et al.*, 2006) and the python optimization toolbox SMARTy (Bekemeyer *et al.*, 2022). For machine learning purposes, the open source frameworks TensorFlow (Martin Abadi *et al.*, 2015) and Scikit-learn (Pedregosa *et al.*, 2011) were used. The purpose of TAU is hereby to solve the RANS equations and the adjoint problem. SMARTy wraps around TAU and enables running the different modes of TAU in the context of the stated optimization problem, that is, running TAU in flow solver mode to evaluate the cost function, and running TAU in adjoint solver mode to compute the gradient $\frac{dI}{d\beta}$. SMARTy also wraps TensorFlow, thereby enabling communication between the machine learning model and the flow solver.

Table 1: The cases selected from the database for field inversions.

Case	Re	Ma	α
T1	2.68×10^6	0.717	2.604°
T2	6.36×10^6	0.742	4.456°
T3	8.79×10^6	0.721	5.669°
T4	10.94×10^6	0.724	5.654°
T5	13.18×10^6	0.724	5.650°
T6	15.32×10^6	0.724	5.145°

3. DATABASE

The goal of this study is to apply the Field Inversion and Machine Learning approach to flows with shock-induced separation. For this purpose, kindly the AIRBUS RWC.01 database was made available to the authors. The AIRBUS RWC.01 data base gathers aerodynamic experimental data acquired in 2016 using the pilot facility of the European Transonic Wind Tunnel (pETW) for a series of 2D airfoil sections.

3.1 Windtunnel Measurements

The RWC.01 database contains pressure tap measurements for the RAE2822 airfoil from a windtunnel campaign covering a Mach number range from 0.2 to 0.96, a Reynolds number range from 2.7×10^6 to 15.7×10^6 and an angle of attack sweep from $-2.5°$ to $13°$. The RAE2822 is a transsonic, rear-loaded airfoil with a rooftop type pressure distribution. The windtunnel width is three times the chord length of the airfoil, and the pressure is measured along the centerline of the airfoil, hence 3D effects are considered negligible. From this large database, we selected multiple cases in which shock-induced separation appears and the predictions of the baseline SA-neg model were considerably off, selecting cases with differing flow conditions on purpose. A list of the selected cases is provided in Table 1.

3.2 *Field Inversion*

For the field inversion, first, the optimal regularization parameter λ must be determined for each case in Table 1 separately. The canonical way (Jäckel, 2021; Singh and Duraisamy, 2016) to do so is to conduct multiple field inversions for different values of λ, then plot the respective two terms of Eq. (3) against each other in a log-log plot, which should yield a so-called *L-Curve* (Hansen, 2000). There the optimal value for λ lies in the corner of the L-shaped curve, balancing prediction improvements and magnitude of the model modification. In the present case, we could not obtain such a curve, possibly due to unknown uncertainties in the wind tunnel measurements or suboptimal convergence in the optimization.

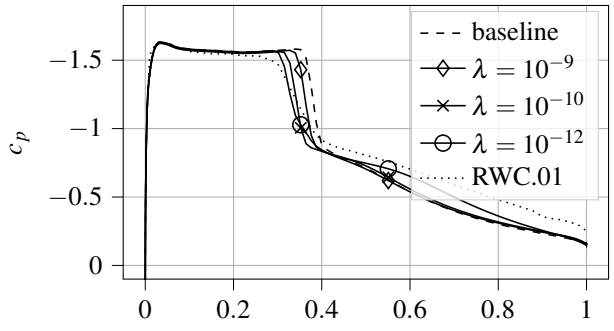

Figure 2: c_p distributions of case T3 from the baseline computation, different λ and from the reference data.

Instead, we resorted to engineering judgement. For a too large λ, the shock location does not change significantly from the baseline solution, and the β-field is not very pronounced. For a too small λ, the shock location is matched well, and the pressure downstream of the shock is also matched better. However, this comes at the cost of a much more pronounced β-field, with stronger gradients and more extrema, which is considered to complicate the machine learning step. Additionally, the results from the pressure sensors downstream of the shock are uncertain. Hence, we choose λ such that

- the shock location is improved, but the pressure downstream is maintained, and
- the β-field does not become overly complex.

Figures 2 and 3 show the effects of different values of λ on c_p and β respectively.

Figure 4 shows for each case the c_p distributions on the upper side of the airfoil for the field inversion result (solid line), the baseline solution (dashed line), and the reference data (dotted line). For case T1, no regularization parameter was found that would improve the results without hurting our engineering considerations. For cases T2 through T6, the shock location is improved considerably throughout, without hurting the engineering considerations.

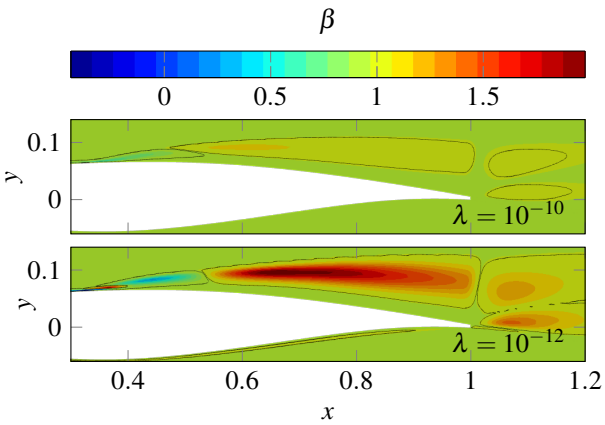

Figure 3: β-fields for different λ, case T3. Note the additional extrema and the wider range of β for $\lambda = 10^{-12}$.

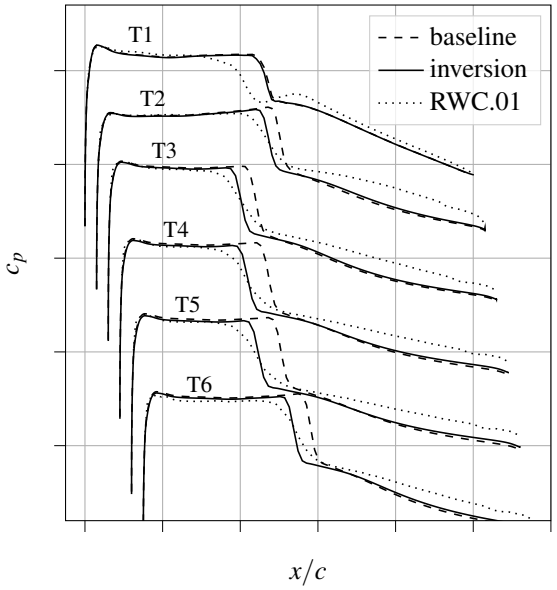

Figure 4: Surface c_p from the baseline RANS computation, field inversion, and reference database for cases T1 - T6.

Figures 5 and 6 show the full flow fields in terms of the pressure coefficient c_p and the Mach number of the field inversion result for case T3 respectively. In both, the oblique shock wave beginning at $x/c \approx 0.4$ is well recognizable. The streamlines shown in Figure 6 emphasize the separation bubble downstream of the shock location.

Finally, Figure 7 shows the resulting β-fields for cases T1, T3 and T6. Cases T2, T4 and T5 are not shown due to their similarity to cases T3 and T6. For cases T2 through T6, β is typically decreased to around $\beta = 0.5$ at the shock location ($x/c \approx 0.4 \ldots 0.6$, depending on the case) and the surrounding areas. This decrease corresponds to

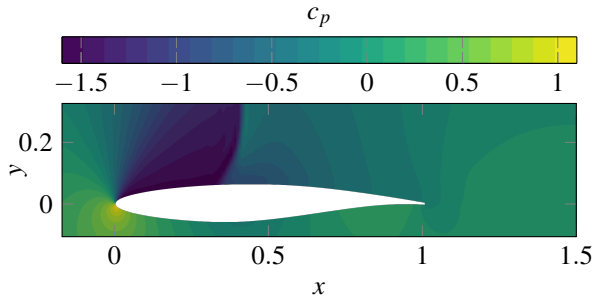

Figure 5: c_p from the field inversion, case T3.

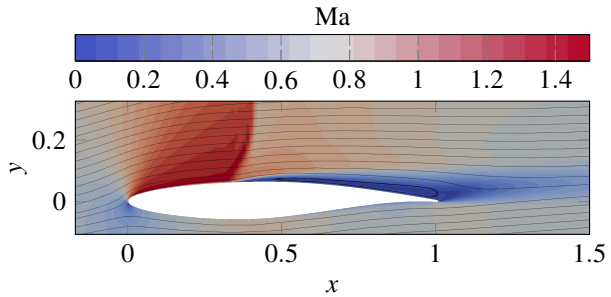

Figure 6: Local Mach number and streamlines from the field inversion results for case T3.

reduced turbulent production, hence a reduced eddy viscosity and therefore the shock location moves upstream. At the beginning of the ensuing separation bubble, turbulent production is still decreased. Then, an area with an increased β, that is, increased turbulent production, follows at the outer edge of the separation bubble. For case T1 however, the flow remains attached, and turbulent production is decreased only slightly along the entire upper surface. As discussed previously, this case was particularly troublesome to find a plausible regularization parameter for, and a relatively high regularization was used.

4. FEATURE SELECTION

With the data gathered in the previous section, the feature selection procedure discussed in Section 2.2 was applied.

4.1 Considered Features

The following features are considered:

1. The normalized transport variable of the SA-neg model $\chi = \tilde{\nu}/\nu$, where ν is the molecular kinematic viscosity.

2. The ratio of the production to the destruction term of the SA-neg model P/D

3. The dimensionless function f_w, part of the destruction term D of the SA-neg model, see (Allmaras *et al.*, 2012).

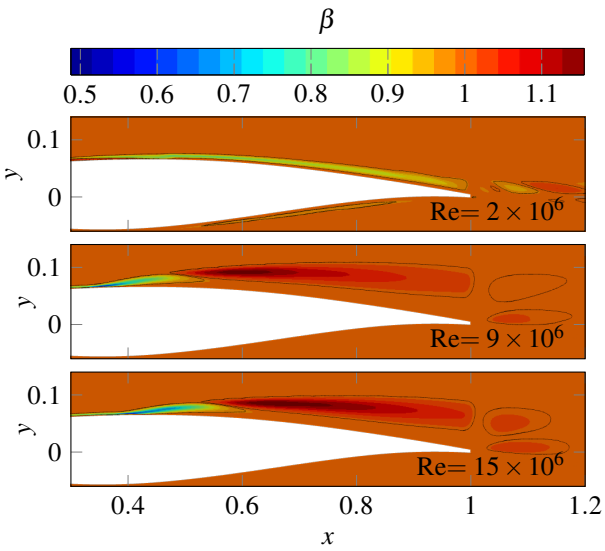

Figure 7: β-fields for cases T1, T3 and T6. Note that the results for cases T2, T4, T5 are very similar to T3 and T6 and hence not depicted.

4. A non-dimensionalized variant of the SA-neg viscosity gradient magnitude $\overline{\nabla\tilde{\nu}} = d/(\nu+\tilde{\nu})|\nabla\tilde{\nu}|$, according to (Ferrero *et al.*, 2020). d is the wall distance.

5. The magnitude of the vorticity tensor, non-dimensionalized $\overline{\Omega} = d^2/(\nu+\tilde{\nu})\Omega$, where Ω is the magnitude of the vorticity $\Omega = \|\Omega_{ij}\|$.

6. The ratio of the local turbulent stresses to the shear stress at the closest wall τ_w (Holland *et al.*, 2019; Jäckel, 2022), $\delta = \mu_t S/(1.5\tau_w)$

7. The ratio of the magnitudes of strainrate and vorticity $S/\Omega = \|S_{ij}\|/\|\Omega_{ij}\|$

8. The normalized Reynolds stress tensor magnitude τ/τ_{ref}, where $\tau_{ref} = \rho(\nu+\tilde{\nu})^2/d^2$ and $\tau = \|\tau_{ij}\|$. Based on (Ferrero *et al.*, 2020).

9. The boundary layer shape factor $H_{12} = \delta^*/\theta$, where δ^* is the displacement thickness, and θ is the momentum thickness of the boundary layer.

10. A measure of the turbulent kinetic energy, $k'_{QCR} = 1.5C_{Cr2}\nu_t\sqrt{2S_{ij}S_{ij}}/(\frac{1}{2}\sum_i u_i^2)$, based on the quadratic constitutive relation (QCR) for SA-neg models, as used by (Volpiani *et al.*, 2021). It is $C_{Cr2} = 2.5$, and u_i are the velocities.

11. The DDES wall shielding function $f_d = 1 - tanh(8r_d^3)$, used before in this or a modified version by (Ferrero *et al.*, 2020; Köhler *et al.*, 2020).

12. The Rotta and Clauser pressure gradient parameter $\beta_{RC} = \delta^*/(\rho u_\tau^2)\,\partial p/\partial s$, where $\dfrac{\partial p}{\partial s}$ is the streamwise pressure gradient.

13. The inner pressure gradient parameter $\Delta p_{s+} = \nu/(\rho u_\tau^3)\,\partial p/\partial s$

Note that not all features fulfill the criteria of being locally available, for example δ, H_{12} and the pressure gradient parameters β_{RC} and Δp_{s+}. These non-local features typically depend on surface quantities and are made available locally by projecting the surface values into the field, that is, at every field point the value of the quantitiy at the closest wall is made available.

4.2 Spearman's Correlation

We now combine the datasets for the field inversions from all cases and compute Spearman's rank correlation coefficient between all of the features and the features and β. Figure 8 shows the resulting correlation matrix. From the

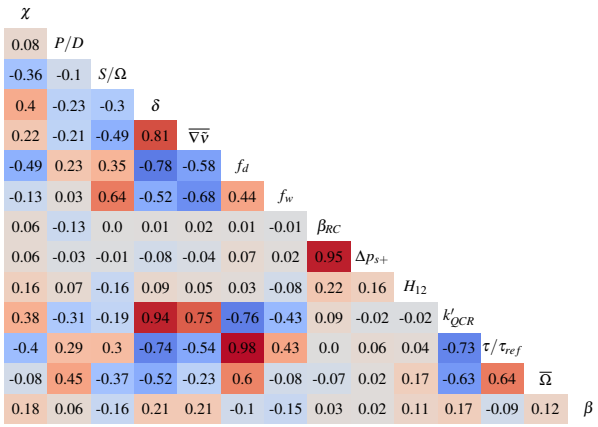

Figure 8: Correlation matrix

correlation matrix, we find that the following features are redundant:

- k'_{QCR} and δ at a correlation of $r_S = 0.94$. Looking at their definitions, this is expected, as both features are proportional to $\nu_t S$. The correlation of k'_{QCR} and β is $r_S = 0.17$, less than the correlation of δ and β at $r_S = 0.21$, hence k'_{QCR} is eliminated.

- f_d and τ/τ_{ref} at a correlation of $r_S = 0.98$. Here, τ/τ_{ref} is eliminated due to its lower correlation with β at $r_S = -0.09$ as opposed to $r_S = -0.1$ for (f_d, β). Deeper insight reveals that both features scale with $\frac{d^2}{\tilde{\nu}}$ and one of $\{S, \sqrt{\frac{du_i}{dx_j}\frac{du_j}{dx_i}}\}$ in relevant parts of the flow, explaining their high correlation.

- β_{RC} and Δp_{s+} at a correlation of $r_S = 0.95$. According to their definitions, both are different scalings of $\partial p/\partial s$, hence this is expected. Both are eliminated, as their correlation with β is very small with $r_S = 0.03$ and $r_S = 0.02$ respectively.

This results into the following subset of features:

$$\{\overline{\Omega}, H_{12}, f_w, f_d, \overline{\nabla\tilde{\nu}}, \delta, \frac{S}{\Omega}, \frac{P}{D}, \chi\} \tag{7}$$

4.3 Sequential Feature Selection

Before running the Sequential Feature Selection, which includes the actual training of a neural network, the data needs to be preprocessed.

4.3.1 Preprocessing

Some of the features values span multiple orders of magnitudes. Applying the natural logarithm to these features makes them more amenable to machine learning by pushing their distribution towards a Gaussian distribution.

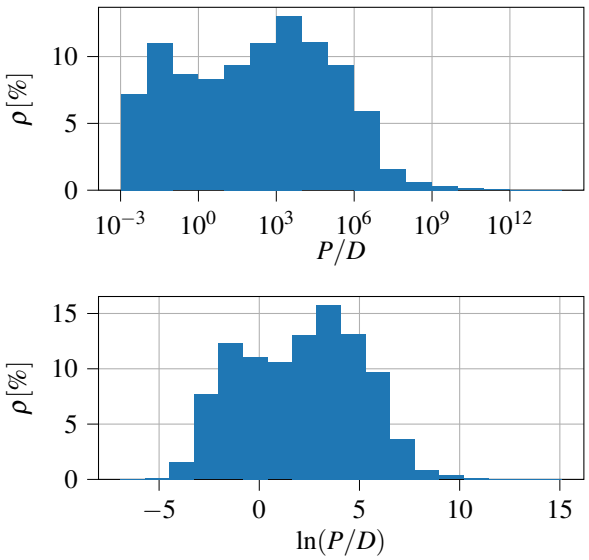

Figure 9: Histograms for P/D (top) and $\ln(P/D)$ (bottom). Note the different scales of the x-axis.

Figure 9 demonstrates the effect this transformation has on the example of the feature P/D. The feature spans orders of magnitudes from $\mathcal{O}(10^{-3})$ to $\mathcal{O}(10^{15})$. Applying the logarithm brings all data to an order of magnitude of $\mathcal{O}(1)$, and an almost Gaussian distribution, which makes the data much easier discernable to the neural network. The features to which the natural logarithm is applied are

$$\{\overline{\Omega}, \overline{\nabla\tilde{\nu}}, \delta, \frac{S}{\Omega}, \frac{P}{D}, \chi\}, \tag{8}$$

Note that this transformation leads to a loss of information, as the logarithm is not defined for negative values which means that affected samples must be removed from the dataset. Most of the features become only

negative when $\tilde{\nu} < 0$. In that case, the modification introduced by the negative Spalart-Allmaras model formulation becomes active. We expect it to be better to leave the turbulence model unchanged in that case for numerical robustness, therefore it is no problem to leave out these samples.

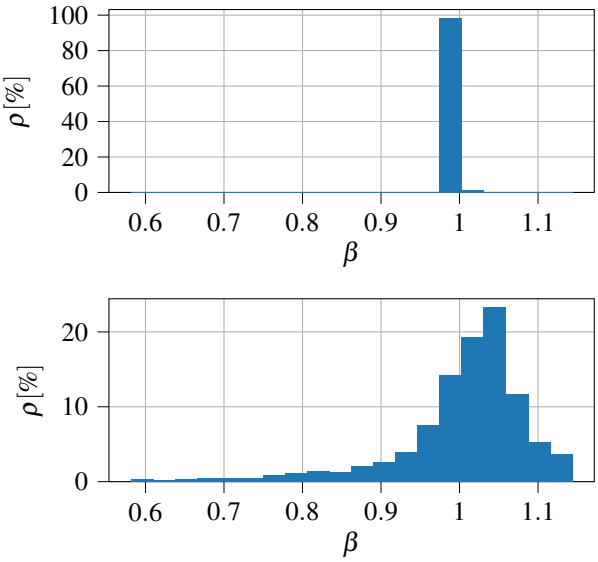

Figure 10: Histograms for the full set of β (top) and the reduced set of β (bottom).

A further step is to reduce the amount of samples where $\beta \approx 1$. This is true for the majority of the samples, see Figure 10, upper diagram. For the purpose of machine learning this is detrimental, as it rewards the ML algorithm to predict a constant $\beta = 1$, as this already decreases the cost function significantly. The procedure hereby is to pick all $n_{\beta \neq 1}$ samples for which $|\beta - 1| > 0.02$ holds, and then to add $0.2 n_{\beta \neq 1}$ of the samples with $|\beta - 1| < 0.02$. These thresholds and factors have been obtained manually by targeting a more Gaussian distribution. According to Figure 10, the distribution of β is now much closer to a Gaussian distribution and therefore better suited for ML training.

These preprocessing steps leave around 7100 samples, about 1% of the initial dataset size.

4.3.2 Procedure

A neural network with two layers and 50 neurons each is used. As activation functions the rectified linear unit (ReLU) is used together with the *Adam* optimizer. Due to the random initialization of neural networks, SFS runs are non-deterministic. To alleviate this, each SFS variant, forward and backward, is run three times. We predefine a range of 3 to 5 for the number of features the SFS should return. For each run, we then assign a score to the returned feature, ranging from 5 for the most important feature to 1 for the least important feature. Then, the scores over the different SFS runs are summed up to get the final score for each feature.

4.3.3 Results

The final scores from SFS as well as the correlations with β, both, for the full ("full data") and the preprocessed ("training data") datasets are listed in Table 2.

Table 2: Selection table for input features to be used for neural network training

Feature	**SFS score**	**Correlation with $\beta(x)$**	
		Full data	Training data
$\ln(\chi)$	30	0.18	0.61
$\ln(\delta)$	18	0.21	0.62
$\ln(\overline{\Omega})$	17	0.12	0.20
$\ln\left(\frac{S}{\Omega}\right)$	8	-0.16	-0.11
H_{12}	8	0.11	0.18
f_d	2	-0.1	0.13
f_w	1	-0.15	0.20
$\ln(\overline{\nabla\tilde{\nu}})$	0	0.21	0.18

By far, $\ln(\chi)$ is considered the most important feature, followed by $\ln(\delta)$ and $\ln(\overline{\Omega})$, both at the same level. With less importance, $\ln(S/\Omega)$ and H_{12} follow. Not much significance is attributed to f_d, f_w and $\ln(\overline{\nabla\tilde{\nu}})$.

5. MACHINE LEARNING

With the results obtained before, finally a neural network can be trained and included in the CFD solver to augment the SA-neg turbulence model. As inputs, we use the features

$$\{\ln(\chi), \ln(\delta), \ln(\overline{\Omega}), \ln(\frac{S}{\Omega}), H_{12}\}, \tag{9}$$

which are the top 5 of the features listed in Table 2. The neural network architecture is optimized using the hyperparameter optimization strategy discussed in (Sabater *et al.*, 2021). The result is a network with 4 layers, $86, 67, 52$ and 40 neurons respectively.

Figure 11 shows the training success. The line indicates perfect predictions, the symbols show the actual predictions, located not far off the line. With the thus-augmented turbulence model, CFD calculations are carried out for the cases that were part of the training dataset, T2 - T6, as well for some additional validation cases, T1 and V1-V3, see Table 3.

In Figure 12, results are shown for multiple training and validation cases. It contains the surface c_p of the

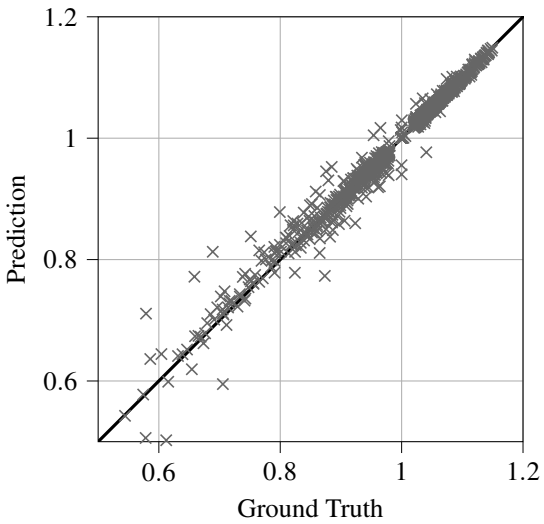

Figure 11: Truth vs Prediction for the trained NN.

Table 3: Validation cases

Case	Re	Ma	α
V1	6.33×10^6	0.721	5.737°
V2	8.76×10^6	0.742	4.420°
V3	15.36×10^6	0.711	5.145°

baseline model (dashed line), the reference data (dotted), the NN-augmented prediction (dash-dotted), and, where applicable, the inversion result (solid line). For cases T2, T3 and V2, the prediction of the shock location is significantly improved, reaching the field inversion solution. For V1, the improvement is less clear, since the shock location is not as clear in the reference data. For cases T1 and V3 no shock-induced separation appears and the shock location is not improved. In both cases however, the prediction is also not worsened vis-â-vis the baseline model. Note that T1 was not part of the training data as the field inversion did not yield usable improvements.

In (Jäckel, 2022), a different approach with the Field Inversion and Machine Learning approach was taken which fits an analytical function instead using the direct FIML (Holland *et al.*, 2019) variant. It uses test case T3 as well however and achieves the same improvement in the prediction of the shock location.

6. SUMMARY

In this study, the Field Inversion and Machine Learning approach was applied to cases with shock-induced separation. For this purpose, an extensive database of windtunnel measurements for the RAE2822 at transonic flow conditions was investigated. This included the

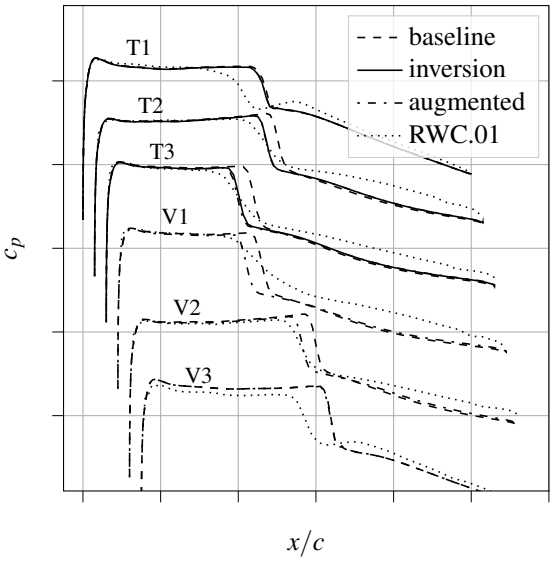

Figure 12: Surface c_p for cases T1 - T3 and V1 - V3. For T1 - T3, the inversion and augmented results are on top of each other.

identification of cases for which the CFD simulations with the SA-neg model deviated significantly from the measured data, and selecting these cases for field inversions. The next step was feature selection, where a list of possible input features for the machine learning step was established and where the features were investigated in terms of their applicability to the present problem by examining their correlations among themselves and with the target term as well as via sequential feature selection. Then, the hyperparameters of the deployed neural network were optimized. The thus optimized network was trained using the previously selected input features to predict the correction term introduced into the turbulence model, and successfully applied to multiple cases, both, cases that were part of the training data and cases that were unseen. To apply the trained model to other testcases with similar flow conditions, a more comprehensive training process based on the FIML Direct approach (Holland *et al.*, 2019) is needed. The approach was already adopted for subsonic flows (Jäckel, 2022) and enables the training process on several testcases simultaneously. In the present work, the focus was on feature selection methods and on first steps for data-driven turbulence modeling in transonic flows.

Future points of research are

- to combine the training data obtained in this study for shock-induced separation with previously obtained data (Jäckel, 2021) for subsonic trailing edge separation and investigate if a common correction model can be found based on the preferred features as determined in this study

- to investigate different correlation coefficients, as Spearman's correlation coefficient only considers two variables at a time, and detects only monotonous relationships.
- to add more features and other transformations than the natural logarithm such as the transformation proposed by (Ling *et al.*, 2016).
- to put more focus on the availability of the features in other CFD solvers, where non-local features such as for example the wall shear stress at the closest wall might not be given.
- to train on several transonic flow testcases for an improved generalizability of the resultung data-driven turbulence model.

ACKNOWLEDGMENTS

This work was funded by the sixth Federal Aeronautical Research Programme Germany in the project DIGIFly - Digital Flight of Air Vehicles under grant number 20X1909A. The authors are grateful to AIRBUS for providing the RWC.01 aerodynamic data base.

REFERENCES

Allmaras, R. S., Johnson, T. F., and Spalart, R. P. (2012). "Modifications and clarifications for the implementation of the Spalart-Allmaras turbulence model". In: *Seventh International Conference on Computational Fluid Dynamics*. ICCFD. Big Island, Hawaii, USA.

Beck, A. and Kurz, M. (2021). *A perspective on machine learning methods in turbulence modeling*. DOI: 10.1002/gamm.202100002.

Bekemeyer, P. *et al.* (2022). "Data-Driven Aerodynamic Modeling Using the DLR SMARTy Toolbox". In: *AIAA AVIATION 2022 Forum*. AIAA 2022-3899. DOI: 10.2514/6.2022-3899.

Cook, P., McDonald, M., and Firmin, M. (1979). "Aerofoil RAE 2822 - Pressure Distributions, and Boundary Layer and Wake Measurements". In: *Experimental Data Base for Computer Program Assessment, AGARD Report AR 138*.

Duraisamy, K. (2021). "Perspectives on machine learning-augmented Reynolds-averaged and large eddy simulation models of turbulence". In: *Phys. Rev. Fluids* 6 (5), p. 050504. DOI: 10.1103/PhysRevFluids.6.050504.

Dwight, R. P. and Brezillon, J. (2006). "Effect of Approximations of the Discrete Adjoint on Gradient-Based Optimization". In: *AIAA Journal* 44.12, pp. 3022–3031. DOI: 10.2514/1.21744.

Ferrero, A., Iollo, A., and Larocca, F. (2020). "Field inversion for data-augmented RANS modelling in turbomachinery flows". In: *Computers and Fluids* 201, p. 104474.

Giles, M. B. and Pierce, N. A. (2000). "An introduction to the adjoint approach to design". In: *Flow, turbulence and combustion* 65, pp. 393–415. DOI: 10.1023/A:1011430410075.

Hansen, P. C. (2000). "The L-Curve and its Use in the Numerical Treatment of Inverse Problems". In: *Computational Inverse Problems in Electrocardiology, ed. P. Johnston, Advances in Computational Bioengineering*. WIT Press, pp. 119–142.

Holland, J. R., Baeder, J. D., and Duraisamy, K. (2019). "Field Inversion and Machine Learning With Embedded Neural Networks: Physics-Consistent Neural Network Training". In: *AIAA Aviation 2019 Forum*. Dallas, Texas, USA. DOI: 10.2514/6.2019-3200.

Jäckel, F. (2021). "Sensitivity Analysis of Discrepancy Terms introduced in Turbulence Models using Field Inversion". In: *New Results in Numerical and Experimental Fluid Mechanics XIII* 151, pp. 625–634. DOI: 10.1007/978-3-030-79561-0_59.

Jäckel, F. (2022). "A Closed-form Correction for the Spalart-Allmaras Turbulence model for Separated Flows". In: *AIAA SciTech 2022 Forum*. San Diego, CA, USA: American Institute of Aeronautics and Astronautics Inc, AIAA. DOI: 10.2514/6.2022-0462.

Köhler, F., Munz, J., and Schäfer, M. (2020). "Data-driven augmentation of rans turbulence models for improved prediction of separation in wall-bounded flows". In: American Institute of Aeronautics and Astronautics Inc, AIAA. DOI: 10.2514/6.2020-1586.

LeCun, Y., Bengio, Y., and Hinton, G. (2015). "Deep learning". In: *Nature* 521, pp. 436–444. DOI: 10.1038/nature14539.

Ling, J., Jones, R., and Templeton, J. (2016). "Machine learning strategies for systems with invariance properties". In: *Journal of Computational Physics* 318, pp. 22–35. DOI: 10.1016/j.jcp.2016.05.003.

Martin Abadi *et al.* (2015). *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*. Software available from tensorflow.org.

Nocedal, J. and Wright, S. (2006). *Numerical Optimization: Springer Series in Operations Research and Financial Engineering*. English. Springer.

Pedregosa, F. *et al.* (2011). "Scikit-learn: Machine learning in Python". In: *Journal of machine learning research* 12.Oct, pp. 2825–2830.

Probst, A. *et al.* (2020). "HPC Requirements of High-Fidelity Flow Simulations for Aerodynamic Applications". In: *Euro-Par 2019: Parallel Processing Workshops*. Göttingen, Germany: Springer International Publishing, pp. 375–387.

Sabater, C., Stürmer, P., and Bekemeyer, P. (2021). "Fast Predictions of Aircraft Aerodynamics using Deep Learning Techniques". In: *AIAA Aviation 2021 Forum*. virtual event. DOI: 10.2514/6.2021-2549.

Schmelzer, M., Dwight, R. P., and Cinnella, P. (2020). "Discovery of Algebraic Reynolds-Stress Models Using Sparse Symbolic Regression". In: *Flow, Turbulence and Combustion* 104, pp. 579–603. DOI: 10.1007/s10494-019-00089-x.

Schwamborn, D., Gerhold, T., and Heinrich, R. (2006). "The DLR TAU-Code: Recent Applications in research and industry". In: *Eccomas CFD 2006 Conference*.

Singh, A. P. and Duraisamy, K. (2016). "Using field inversion to quantify functional errors in turbulence closures". In: *Physics of Fluids* 28, p. 045110. DOI: 10.1063/1.4947045.

Singh, A. P., Matai, R., *et al.* (2017). "Data-driven augmentation of turbulence models for adverse pressure gradient flows". In: *23rd AIAA Computational Fluid Dynamics Conference*. AIAA 2017-3626. DOI: 10.2514/6.2017-3626.

"Spearman Rank Correlation Coefficient" (2008). In: *The Concise Encyclopedia of Statistics*, pp. 502–505. DOI: 10.1007/978-0-387-32833-1_379.

Volpiani, P. S. *et al.* (2021). "Machine learning-augmented turbulence modeling for RANS simulations of massively separated flows". In: *Physical Review Fluids* 6 (6), p. 064607. DOI: 10.1103/PhysRevFluids.6.064607.

Weatheritt, J. and Sandberg, R. (2016). "A novel evolutionary algorithm applied to algebraic modifications of the RANS stress–strain relationship". In: *Journal of Computational Physics* 325, pp. 22–37. DOI: 10.1016/j.jcp.2016.08.015.

Wu, J. L., Xiao, H., and Paterson, E. (2018). "Physics-informed machine learning approach for augmenting turbulence models: A comprehensive framework". In: *Physical Review Fluids* 3 (7), p. 074602. DOI: 10.1103/PhysRevFluids.3.074602.

Zhang, Z. J. and Duraisamy, K. (2015). "Machine Learning Methods for Data-Driven Turbulence Modeling". In: AIAA 2015-2460. DOI: 10.2514/6.2015-2460.