Gradient-Based Shape Optimization for Unsteady Turbulent Simulations Using Dynamic Closures

Krzysztof J. Fidkowski*
University of Michigan, Ann Arbor, MI, 48188

This paper presents a method for gradient-based shape optimization using unsteady models of turbulent flowfields, for which forward simulations are already expensive and adjoint calculations diverge or require costly regularization. The proposed method targets an objective and constraints computed from the time-averaged flowfield and does not store the forward time history or compute an unsteady adjoint. It instead relies on the field-inversion, machine-learning (FIML) approach, in which a correction field modifies the production term in a Reynolds-averaged Navier-Stokes (RANS) model of the flow. Steady, adjoint-based, field inversion yields this correction field, and a neural-network model is trained to reproduce this field from local flow quantities. Gradient-based shape optimization is performed using the corrected RANS model, which includes a linearization of the correction field calculation for accurate gradients. The complete design optimization loop consists of iterations of unsteady simulation, FIML, and steady optimization. Low and moderate Reynolds number airfoil optimization problems demonstrate the performance of the proposed method, and comparisons to RANS-alone designs illustrate the importance of accounting for the unsteady flow effects in the optimization.

I. Introduction

Turbulent flows exist in many fluid systems, including the aerodynamics of aerospace vehicles. Many techniques have been developed to simulate such flows, ranging from completely ignoring viscous effects, for example through potential-flow or Euler equations, to resolving every turbulent scale via direct numerical simulation (DNS). In between lie steady-state methods based on Reynolds averaging (RANS), and unsteady methods such as large-eddy and detached-eddy simulations (LES, DES). Whereas steady-state methods are generally adequate for vehicles at "on-design" conditions, high-fidelity simulations at off-design conditions, often characterized by regions of separated flow, generally require unsteady methods. This latter class of flows provides the setting for the present work, which addresses the design of aerodynamic shapes at off-design conditions.

Unsteady models of turbulent flow complicate shape optimization in at least two ways. First, the simulations are much more expensive than steady models, so that each function evaluation, usually a statistic such as a time-averaged output, requires orders of magnitude more computational time/resources. This limits the applicability of many-query studies, such as optimization, particularly gradient-free methods that rely on numerous function evaluations [1]. Second, the chaotic nature of unsteady turbulent simulations prevents the application of linear sensitivity techniques for evaluation of gradients. Adjoint methods, the hallmark of shape optimization [2–6], cannot be directly applied, as the unsteady adjoint equations are unstable in such systems [7] and require expensive regularization techniques to provide meaningful answers [8–13]. Furthermore, unsteady adjoints come with massive storage and computation requirements that become impractical for forward simulations, which already stress computational resources [14]. For these two reasons, the present work is not based on unsteady adjoint methods.

Unsteady adjoints are useful when computing sensitivities of deterministic events, such as maneuvers, gust interactions, or short-duration aeroelastic events [15–23]. However, for turbulent flows, they provide perhaps too much information: the sensitivity of an output to a flow residual at a single point in space and time loses significance when that particular flow state may not appear in another simulation that follows a different trajectory [24]. Instead, of engineering interest are generally only statistical quantities, such as time averages of outputs. If a steady-state model were able to yield these quantities with sufficient accuracy, it would be much more amenable to gradient-based optimization.

Many steady-state turbulence models exist, most based on Reynolds averaging of the Navier-Stokes equations. However, as discussed previously, these models often do not provide sufficient accuracy relative to unsteady models at

*Professor, Department of Aerospace Engineering, AIAA Associate Fellow.

off-design conditions. Indeed, the creation of a general turbulence model that is accurate across multiple flow regimes continues to be an elusive quest [25]. Presently, we forego generality and create multiple steady-state models, each specific to the unsteady case being considered. We only use the model to enable an efficient calculation of the derivatives required for optimization, instead of trying to apply the model to vastly different flow regimes.

The turbulence modeling approach we take here is based on the idea of field inversion and machine learning (FIML) [26–32]. The idea of FIML is to start with an existing steady-state model and to correct it via a PDE-level correction factor on one or more of the turbulence modeling terms. The unsteady simulation, here in lieu of experimental or other external data, provides the truth solution for the inverse problem for this correction factor, and a machine-learning approach maps local flow features to the correction factor. The result is a corrected steady-state model that can be used on different meshes or shapes. Whereas previous works have used FIML as a general turbulence modeling strategy with mixed generalizability success [25], in this work we only rely on FIML's local accuracy, at/near the conditions for which it was trained. This relaxation of expectations then simplifies aspects of the FIML approach, particularly in the amount of training data, size of the machine learning neural network, and the generality of the inputs.

The proposed unsteady optimization approach, illustrated in Figure 1, consists of iterative turbulence model calibration, through FIML and a *small* number of unsteady simulations, coupled with steady-state optimization using the corrected turbulence models. The number of unsteady simulations required is much lower than the function evaluations

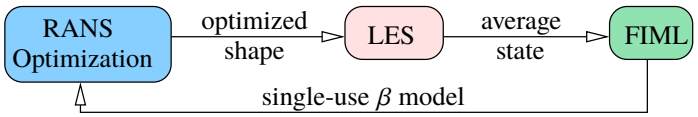


Fig. 1 The proposed unsteady optimization approach using dynamically-corrected RANS models.

demanded by any gradient-free or even gradient-based method, since the optimization is offloaded to the steady-state model. The unsteady simulations only provide model training data, and as the model is often accurate in the vicinity of training, only a few unsteady evaluations (2-4 in the present results) are typically required. However, as will be shown in the results, the accuracy improvements of the corrected model can make a difference in the optimized shapes, particularly in cases where the uncorrected model looses applicability.

The outline for the remainder of this paper is as follows. Section II presents the numerical approach used in this work, a discontinuous finite-element discretization for both the primal and adjoint problems. Section III presents the design parametrization for airfoil shapes, and the gradient-based optimization method. Sections IV and V discuss the field-inversion and machine learning approaches, which are used in the unsteady optimization approach that is summarized in Section VI. Finally, Section VII presents results of several unsteady airfoil optimizations, and Section VIII concludes with a summary and a discussion of future directions.

II. Discretization

A. Governing Equations

We present the adaptive approach in the context of a discontinuous Galerkin (DG) finite element discretization [33–35]. We consider a system of unsteady partial differential equations in conservative form,

$$\frac{\partial \mathbf{u}}{\partial t} + \nabla \cdot \vec{\mathbf{F}}(\mathbf{u}, \nabla \mathbf{u}) + \mathbf{S}(\mathbf{u}, \nabla \mathbf{u}) = \mathbf{0}, \tag{1}$$

where $\mathbf{u} \in \mathbb{R}^s$ is the s-component state vector, $\vec{\mathbf{F}} \in \mathbb{R}^{\dim \times s}$ is the flux vector, dim is the spatial dimension, and $\mathbf{S}$ is the source term arising from the turbulence model, in this work the Reynolds-averaged Navier-Stokes (RANS) equations with the Spalart-Allmaras (SA) closure [36, 37]. A detailed exposition of the equations and closure relations of the model as used presently can be found in previous work [37, 38].

Relevant to this work is the eddy viscosity equation, which contains the turbulence production term that will be modified through a multiplicative correction factor, β,

$$\frac{\partial(\rho\tilde{\nu})}{\partial t} + \nabla \cdot (\rho \vec{\tilde{\nu}} \tilde{\nu}) - \frac{1}{\sigma}\nabla \cdot [\rho(\nu + \tilde{\nu} f_n)\nabla\tilde{\nu}] = -\frac{1}{\sigma}(\nu + \tilde{\nu} f_n)\nabla\rho \cdot \nabla\tilde{\nu} + \frac{c_{b2}}{\sigma}\rho\nabla\tilde{\nu} \cdot \nabla\tilde{\nu} + \boxed{\beta} P - D. \tag{2}$$

In this equation, ρ is the density, $\vec{\mathrm{v}}$ is the velocity, $\tilde{\nu}$ turbulence working variable, f_n is a function that is active for $\tilde{\nu} < 0$, c_{b2}, σ are model constants, P is the turbulence production function, and D is the turbulence destruction function. Scaling P by β affects the entire solution as the eddy viscosity equation is coupled to the conservation equations.

The SA model uses the distance, d, to the closest wall in several closure relations. Modifications to d, usually in the form of limits on its maximum value, yield models for unsteady, detached-eddy simulations (DES) [39]. In the present work, we implement a very simple DES model for unsteady high-Reynolds number simulations,

$$d = d_{\max} \tanh(d/d_{\max}), \tag{3}$$

where $d_{\max}$ is a prescribed distance, set to 2% of the chord length for airfoil simulations. In contrast to most DES models, which relate the limits on d to the mesh size, this choice eliminates complications of grid size effects on the unsteady solution. Moreover, we are not overly concerned with the model accuracy, as our simulations are at present two-dimensional. Instead, the simple DES model brings in unsteadiness that is representative of the manner in which actual three-dimensional DES models yield chaotic flow or limit-cycle oscillations that preclude an unsteady adjoint solution for optimization. For the purpose of optimization, we assume that the simple-model DES solutions are higher-fidelity than the steady RANS solutions.

B. The Discontinuous Galerkin Method

The unsteady optimization strategy presented in this work is not specific to the spatial or temporal discretization. For the results, we use a discontinuous Galerkin (DG) finite-element spatial discretization [40], with the Roe [41] convective flux and the second form of Bassi and Rebay (BR2) [42] for the viscous treatment. The state is approximated on an unstructured mesh of non-overlapping elements using polynomials of order p. Following a finite-element weak formulation and a choice of polynomial basis, the semi-discretized form of the equations is

$$\mathbf{M}\frac{d\mathbf{U}}{dt} + \mathbf{R}(\mathbf{U}) = \mathbf{0}, \tag{4}$$

where $\mathbf{U} \in \mathbb{R}^N$ is the discrete state vector, N is the total number of unknowns, $\mathbf{R}(\cdot) \in \mathbb{R}^{N\times N}$ is the nonlinear spatial residual, and $\mathbf{M} \in \mathbb{R}^{N\times N}$ is the block-element sparse mass matrix. For steady simulations, the time derivative term drops out, although pseudo-time continuation remains in the solver to drive the steady residual to zero [43]. The solver consists of a Newton-Raphson method with the generalized minimum residual (GMRES) [44] linear solver, preconditioned by an element-line Jacobi smoother with a coarse-level ($p = 1$) correction [35, 45]. For unsteady simulations, we use a third-order modified extended backward difference formula [46] applied to the the semi-discrete form.

C. The Discrete Adjoint

For a scalar output computed from the state vector, $J(\mathbf{U})$, the discrete steady adjoint vector, $\mathbf{\Psi} \in \mathbb{R}^N$, is the local sensitivity of J to perturbations in the steady residual, $\mathbf{R}$ [47]. Linearization of the residual and output shows that the adjoint satisfies the following linear equation,

$$\left(\frac{\partial \mathbf{R}}{\partial \mathbf{U}}\right)^T \mathbf{\Psi} + \left(\frac{\partial J}{\partial \mathbf{U}}\right)^T = \mathbf{0}. \tag{5}$$

This equation is solved using the same preconditioned-GMRES method used in the primal solver. The adjoint vector can be used to efficiently compute PDE-constrained sensitivities of the output with respect to parameters, $\mathbf{x}$, of the problem. In general, if both the residuals and the output depend on $\mathbf{x}$, the PDE-constrained sensitivity of the output with respect to the parameter vector is

$$\frac{dJ}{d\mathbf{x}} = \frac{\partial J}{\partial \mathbf{x}} + \mathbf{\Psi}^T \frac{\partial \mathbf{R}}{\partial \mathbf{x}}. \tag{6}$$

In this work, $\mathbf{x}$ consists of shape and operational parameters, and derivatives of J and $\mathbf{R}$ with respect to $\mathbf{x}$ are evaluated using finite-differences. This poses no computational bottleneck in the present study with a small number of parameters. For larger numbers of parameters, analytical or algorithmic differentiation methods can be applied to increase the efficiency of this calculation.

III. Design Optimization

We consider trimmed optimization problems, in which we distinguish between shape design parameters, $\mathbf{x}^{\text{des}} \in \mathbb{R}^{n_{\text{des}}}$, and trim parameters, $\mathbf{x}^{\text{trim}} \in \mathbb{R}^{n_{\text{trim}}}$, concatenated into $\mathbf{x} = [\mathbf{x}^{\text{des}}; \mathbf{x}^{\text{trim}}] \in \mathbb{R}^{n_{\text{par}}}$. Formulated as a minimization statement for a scalar output J, the problem reads

$$\begin{aligned} \min_{\mathbf{x}^{\text{des}}} \quad & J(\mathbf{U}, \mathbf{x}) \\ \text{s.t.} \quad & \mathbf{R}(\mathbf{U}, \mathbf{x}) = \mathbf{0} \\ & \mathbf{R}^{\text{trim}} \equiv \mathbf{J}^{\text{trim}}(\mathbf{U}, \mathbf{x}) - \bar{\mathbf{J}}^{\text{trim}} = \mathbf{0} \end{aligned} \tag{7}$$

where $\mathbf{J}^{\text{trim}} \in \mathbb{R}^{n_{\text{trim}}}$ is the vector of trim outputs, and $\bar{\mathbf{J}}^{\text{trim}} \in \mathbb{R}^{n_{\text{trim}}}$ is the vector of target trim values. In this work, we consider $n_{\text{trim}} = 1$: the lift coefficient, c_ℓ, trimmed by angle of attack, α.

A. Shape Optimization with Concurrent Trimming

To optimize the design, we use a gradient-based method: the Broyden-Fletcher-Goldfarb-Shanno (BFGS) [48] algorithm with a backtracking line search. A notable feature of the implementation is concurrent trimming and optimization with a constrained gradient calculation. A given initial design, $\mathbf{x}_0^{\text{des}}$, is first trimmed using multiple solver iterations, where at each iteration the trim parameters are updated according to [49]

$$\mathbf{x}_{k+1}^{\text{trim}} = \mathbf{x}_k^{\text{trim}} + \Delta\mathbf{x}_k^{\text{trim}}, \qquad \Delta\mathbf{x}_k^{\text{trim}} = \left[\frac{d\mathbf{J}^{\text{trim}}}{d\mathbf{x}^{\text{trim}}}\right]_k^{-1} \left(\bar{\mathbf{J}}^{\text{trim}} - \mathbf{J}^{\text{trim}}(\mathbf{U}_k, \mathbf{x}_k)\right), \tag{8}$$

The PDE-constrained sensitivity matrix $\frac{d\mathbf{J}^{\text{trim}}}{d\mathbf{x}^{\text{trim}}} \in \mathbb{R}^{n_{\text{trim}} \times n_{\text{trim}}}$ is computed using adjoints of the trim outputs, and a user-specified trim tolerance, $\delta\mathbf{J}^{\text{trim}}$, serves as the termination criterion.

Following the initial trim, the shape optimization begins. Each design evaluation consists of a forward solution for the state $\mathbf{U}_k$, and adjoint solutions for the objective, J, and trim, $\mathbf{J}^{\text{trim}}$, outputs. The adjoints yield PDE-constrained sensitivities of all outputs with respect to the design and trim parameters. The trimmed gradient of the objective with respect to the design parameters is then

$$\left.\frac{dJ}{d\mathbf{x}^{\text{des}}}\right|_{\text{trim}} = \frac{dJ}{d\mathbf{x}^{\text{des}}} + \frac{dJ}{d\mathbf{x}^{\text{trim}}}\frac{d\mathbf{x}^{\text{trim}}}{d\mathbf{x}^{\text{des}}}, \qquad \frac{d\mathbf{x}^{\text{trim}}}{d\mathbf{x}^{\text{des}}} = -\left[\frac{d\mathbf{J}^{\text{trim}}}{d\mathbf{x}^{\text{trim}}}\right]^{-1}\frac{d\mathbf{J}^{\text{trim}}}{d\mathbf{x}^{\text{des}}}, \tag{9}$$

where the expression for $\frac{d\mathbf{x}^{\text{trim}}}{d\mathbf{x}^{\text{des}}}$ follows from the requirement that changes in the design and trim parameters, $\delta\mathbf{x}$, must satisfy $\frac{d\mathbf{J}^{\text{trim}}}{d\mathbf{x}}\delta\mathbf{x} = \mathbf{0}$. The gradient of the objective output as given in Eqn. 9 is then used in the BFGS algorithm.

B. Line Search

During the BFGS line search, each objective function evaluation accounts for a linearized trim parameter change and includes an off-trim correction. Let $\mathbf{p}$ be the design-space search direction, and μ the current line-search step length. Prior to running the flow solver, the design and trim parameters are modified according to

$$\mathbf{x}^{\text{des}}(\mu) = \mathbf{x}_k^{\text{des}} + \Delta\mathbf{x}^{\text{des}}, \quad \Delta\mathbf{x}^{\text{des}} = \mu\mathbf{p}, \quad \mathbf{x}^{\text{trim}}(\mu) = \mathbf{x}_k^{\text{trim}} + \frac{d\mathbf{x}^{\text{trim}}}{d\mathbf{x}^{\text{des}}}\Delta\mathbf{x}^{\text{des}}, \tag{10}$$

where the purpose of the $\mathbf{x}^{\text{trim}}$ modification is to maintain linearized trim conditions. The corresponding flow solution $\mathbf{U}(\mu)$ may not be exactly trimmed, as the state and outputs can be nonlinear, while the trim modification arises from linearizations about the zero step-length state and design. The objective calculation then includes an off-trim correction,

$$J(\mu) = J(\mathbf{U}(\mu), \mathbf{x}(\mu)) + \left.\frac{dJ}{d\mathbf{x}^{\text{trim}}}\right|_{\mathbf{U}(\mu),\mathbf{x}(\mu)} \Delta\mathbf{x}^{\text{trim}}(\mu), \tag{11}$$

where the trim parameter change is computed from Eqn. 8 using $\mathbf{U}(\mu), \mathbf{x}(\mu)$.

The step length calculated from the line search, combined with the design-space search direction, determines the change in the design parameters. In addition, following the line search, the trim parameters are modified according to both Eqn. 10, for the linear modification, and Eqn. 8, for the nonlinear correction, with the latter using the trim sensitivity matrix already available from the last line-search iteration.

C. Airfoil Parametrization

We consider two parametrizations for airfoil shape design problems. The first is a continuous generalization of the 4-digit series from the National Advisory Committee for Aeronautics (NACA), modified for zero trailing-edge gap, in which the camber and thickness functions are

$$z_c(x) = \frac{z_{c,\max}}{(1-z_{c,\text{loc}})^2}\left[(1-2z_{c,\text{loc}})+2z_{c,\text{loc}}x-x^2\right](x<z_{c,\text{loc}})+\frac{z_{c,\max}}{z_{c,\text{loc}}^2}\left[2z_{c,\text{loc}}x-x^2\right](x\geq z_{c,\text{loc}}), \quad (12)$$

$$t(x) = t_{\max}\left(0.2969\sqrt{x}-0.126x-0.3516x^2+0.2843x^3-0.1036x^4\right), \quad (13)$$

where $z_{c,\max}$ is the maximum camber, i.e. the first NACA series digit divided by 100, $z_{c,\text{loc}}$ is the location of maximum camber, i.e. the second digit divided by 10, and $t_{\max}$ is the maximum thickness, i.e. the last two digits divided by 100. These expressions assume a unit chord airfoil with $0 \leq x \leq 1$. In the present study, we allow for continuous variation of the airfoil parameters.

The second parametrization allows for more flexibility in the camber and thickness profiles. It consists of polynomial expressions for both profiles, with bounding multiplicative envelopes,

$$z_c(x) = x(1-x)\left[c_0+c_1x+\ldots+c_{p_c}x^{p_c}\right], \quad (14)$$

$$t(x) = a\ \text{abs}_{\text{smooth}}\left\{\sqrt{x}(1-x)\left[1+t_1x+\ldots+t_{p_t}x^{p_t}\right],\ 0.2x(1-x)\right\} \quad (15)$$

where c_i are the coefficients of the order p_c camber polynomial, t_i are the coefficients of the order p_t thickness polynomial, and the smooth absolute value function is defined for positive b as

$$\text{abs}_{\text{smooth}}(a,b) = \begin{cases} |a| & \text{if } |a| \geq b \\ (a^2+b^2)/(2b) & \text{otherwise} \end{cases} \quad (16)$$

The coefficient a enforces a constant airfoil area, A,

$$\int_0^1 t(x)\,dx = A. \quad (17)$$

D. Example Steady Optimization

To demonstrate the geometry parametrization and optimization algorithm, we consider the design of a minimum-drag airfoil at transonic cruise conditions, $M = 0.8, Re = 10^6, c_\ell = 0.5$. Five parameters define the camber and four define the thickness, according to Eqn. 14 and Eqn. 15. The airfoil area is set to $A = 0.06$ (unit chord). An initial symmetrical airfoil shape is constructed by setting all 9 camber/thickness parameters to 0. The computational mesh consists of 5390 quadrilaterals, curved to follow the geometry using an order $q = 3$ reference-to-global mapping. This airfoil is then trimmed to $c_\ell = 0.5$ using an adjoint-based process [49] with a pitch mesh deformation creating the angle of attack change. Artificial-viscosity shock capturing [50] prevents spurious oscillations in the $p = 2$ solution. The optimization then proceeds using the concurrent optimization and trimming algorithm described in Section III.A.

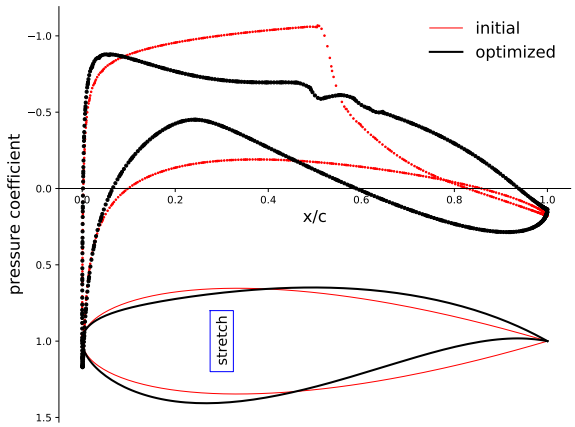
Figure 2 shows the results of the optimization. The initial airfoil exhibits a strong shock on the upper surface at approximately 55% chord. In the optimized design, this shock is virtually eliminated through a flattened upper surface and increased camber near the trailing edge, resulting in a larger aft-loading characteristic of a supercritical airfoil. The output and parameter history plots show that the minimum drag value is achieved rather quickly, even though the shape and angle of attack keep changing as the load is redistributed on the airfoil. This occurs because after the first few optimization iterations, the remaining shock is weak and its contribution to the drag is small.

IV. Field Inversion

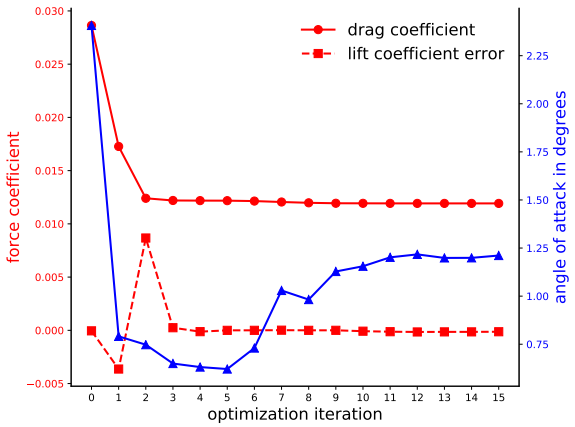
A. Unsteady Solution Averaging

Let $\bar{\mathbf{U}}$ be a statistically-steady flow state computed from the time-average of the unsteady simulation after initial transients,

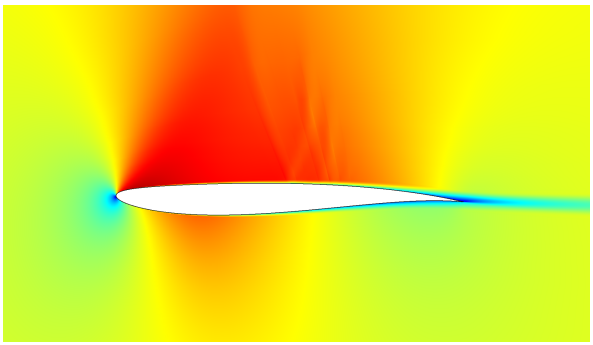
$$\bar{\mathbf{U}} = \frac{1}{T_f - T_i}\int_{T_i}^{T_f} \mathbf{U}(t)\,dt, \quad (18)$$



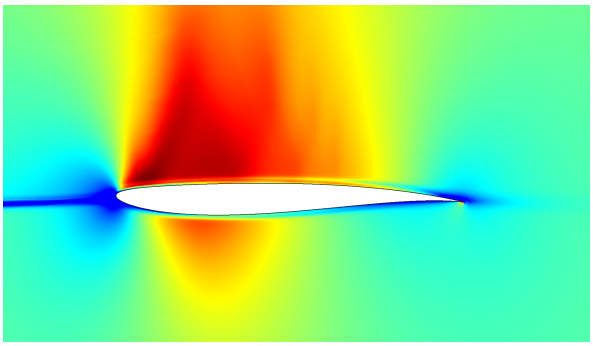
(a) shape and pressure coefficient



(b) output and parameter history



(c) Mach number contours, 0 – 1.3



(d) drag adjoint, x-momentum conservation component

Fig. 2 Steady transonic airfoil shape optimization example: drag coefficient minimization at $M = 0.8, Re = 10^6, c_\ell = 0.5$. Field plots at optimized design.

where T_i is the start time, taken sufficiently large to minimize startup transient effects, and T_f is the final time, taken sufficiently large to yield an adequate statistical mean. The goal of field inversion is to determine a correction field, which enters into the governing equations at the PDE level, to a steady-state model such that the corrected steady-state solution reproduces $\bar{\mathbf{U}}$.

The proximity of the corrected solution to $\bar{\mathbf{U}}$ is measured by the field-inversion error, $\mathcal{E}$, which could be the error in an engineering quantity such as lift or drag, a weighted combination of errors in scalar quantities, an L_2 norm of the entire state error, an L_2 norm of a surface stress distribution error, etc. Presently, we consider the latter,

$$\mathcal{E} = \frac{1}{2}\int_{\text{airfoil}} \|\boldsymbol{\sigma}(\mathbf{U})\cdot\vec{n} - \boldsymbol{\sigma}(\bar{\mathbf{U}})\cdot\vec{n}\|^2\, ds, \tag{19}$$

where $\boldsymbol{\sigma}$ is the stress tensor and $\vec{n}$ is the unit normal vector on the airfoil surface. We note that the entire time-averaged flowfield is available and could also be used in the error definition. However, by focusing on the surface stress distribution, we have observed better matching of force outputs and their sensitivities.

B. Correction-Field Inversion

In this work, the correction is a scalar field, $\beta(\vec{x})$, that multiplies the production term of the SA turbulence model. The nominal value of this correction field is $\beta = 1$. The discrete representation of this field depends on the discretization, and in this work, $\beta(\vec{x})$ is approximated by a $p = 1$ Lagrange DG basis on each element, even for $p > 1$ state approximations. Higher approximation orders for $\beta(\vec{x})$ were found to increase the size of the inverse problem without significantly improving accuracy. Let $\boldsymbol{\beta}$ be the vector of basis coefficients used in the approximation of $\beta(\vec{x})$, numbering $3N_e$ and $4N_e$ values for triangles and quadrilaterals, respectively. The use of a Lagrange basis makes the nominal $\boldsymbol{\beta}$ a vector of all ones. The field inversion then becomes a discrete optimization problem,

$$\begin{aligned} \min_{\boldsymbol{\beta}} \quad & J^{\text{inv}} \equiv \mathcal{E}(\mathbf{U}(\boldsymbol{\beta})) + \frac{\gamma}{2}(\boldsymbol{\beta}-1)^T\mathbf{M}(\boldsymbol{\beta}-1) \\ \text{s.t.} \quad & \mathbf{R}(\mathbf{U},\boldsymbol{\beta}) = \mathbf{0} \end{aligned} \tag{20}$$

We have dropped the dependence of the residual on $\mathbf{x}$, as the design and trim parameters remain fixed during inversion. The additional term in the minimization problem is a continuous Tikhonov regularization that makes the inverse problem well-posed by penalizing large deviations of the correction field from the nominal value of one. $\mathbf{M}$ is the mass matrix from the spatial discretization, and γ is a small user-prescribed parameter that needs to be sufficiently large to prevent ill-conditioning but not too large so as to affect the minimization of $\mathcal{E}$. A broad range of values was typically found to be acceptable, and for the normalized problems considered here, $\gamma = 10^{-7}$ was used.

To solve Eqn. 20, we use a simple steepest descent minimization algorithm, in which the PDE-constrained gradient of the objective, J^{inv}, with respect to $\boldsymbol{\beta}$ is evaluated using an adjoint,

$$\frac{dJ^{\text{inv}}}{d\boldsymbol{\beta}} = \left(\boldsymbol{\Psi}^{\mathcal{E}}\right)^T \frac{\partial\mathbf{R}}{\partial\boldsymbol{\beta}} + \gamma\mathbf{M}(\boldsymbol{\beta}-1), \quad \left(\frac{\partial\mathbf{R}}{\partial\mathbf{U}}\right)^T\boldsymbol{\Psi}^{\mathcal{E}} + \left(\frac{\partial\mathcal{E}}{\partial\mathbf{U}}\right)^T = \mathbf{0}. \tag{21}$$

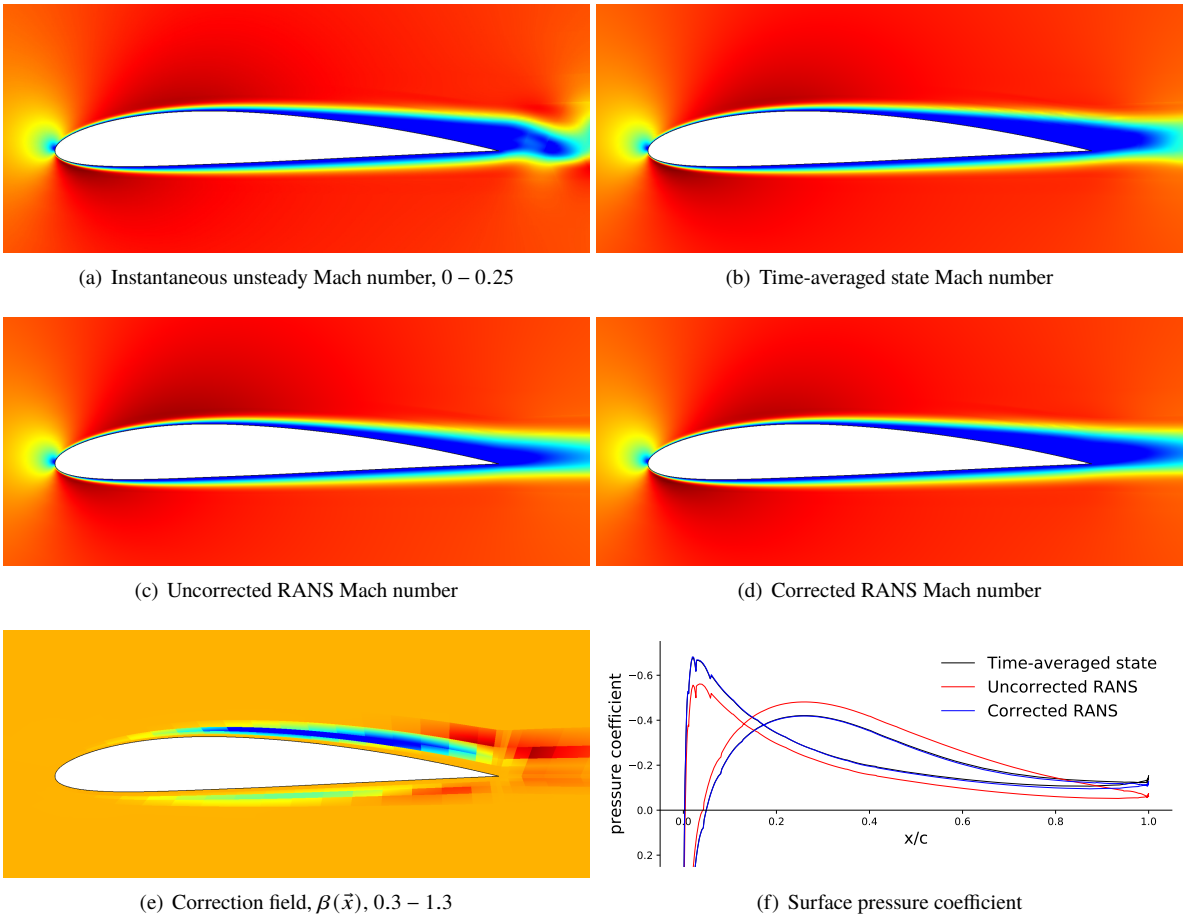
The linearization of the residual with respect to the correction, $\frac{\partial\mathbf{R}}{\partial\boldsymbol{\beta}}$, is local to the elements: by virtue of the DG discretization and the fact that the correction enters via a residual source term, coefficients of $\boldsymbol{\beta}$ for an element affect the discrete residuals only on that element. Therefore, in this work, $\frac{\partial\mathbf{R}}{\partial\boldsymbol{\beta}}$ is evaluated efficiently using finite differences. This requires only a small number of residual evaluations (e.g. 3 for triangles), as derivatives on all elements can be calculated concurrently.

C. Example

To demonstrate the accuracy of the RANS correction, we consider a NACA 3412 airfoil at $M = 0.2$, $Re = 10^4$, and $\alpha = 0$. This low Reynolds-number flow exhibits unsteady behavior that prevents the use of direct unsteady adjoints for time-averaged outputs [51]. The case is solved using $p = 2$ solution approximation on a mesh of 1978 $q = 3$ quadrilateral elements. A third-order modified backwards difference scheme (MEBDF) [46] is used to advance the unsteady cases in time, using a time step of $\Delta t = .05c/U_\infty$, where c is the chord length and U_∞ is the free-stream speed. After a transient, a time horizon of $T_f - T_i = 30c/U_\infty$ is used to calculate the average state according to Eqn. 18.

RANS models are designed for high Reynolds-number turbulent flows, so we should not expect a good agreement between unsteady results and a standard RANS result. This is indeed the case, as shown in Figure 3. The uncorrected

RANS solution exhibits a smaller wake and a different pressure distribution compared to the averaged unsteady result. With the correction to the production term, however, the RANS model can be modified to quite-well match the averaged unsteady result. Figure 3 shows that the Mach number profile and pressure distribution of the corrected RANS result are nearly identical to the time-averaged unsteady result. The field inversion in this case is done using the stress distribution error function, Eqn. 19. The $\beta(\vec{x})$ plot indicates reduced turbulent production on the boundary layer edges, particularly over the upper surface, and increased production towards the trailing edge and wake.



(a) Instantaneous unsteady Mach number, 0 – 0.25

(b) Time-averaged state Mach number

(c) Uncorrected RANS Mach number

(d) Corrected RANS Mach number

(e) Correction field, $\beta(\vec{x})$, 0.3 – 1.3

(f) Surface pressure coefficient

Fig. 3 Field-inversion results for unsteady flow around a NACA 3412 airfoil at $M = 0.2, Re = 10^4, \alpha = 0$.

V. Machine Learning

A. Overview

The correction field obtained from the solution of the inverse problem presented in Section IV makes the solution of the corrected RANS equations match the unsteady average for a particular geometry and mesh. In design optimization, we would like to consider changes to the geometry and trim parameters, which lead to different meshes and states. The correction field then also needs to change, in a manner generally beyond a simple mapping, as new geometry or state features may require local changes to the correction.

One approach to realizing consistent changes to the correction field is to make this field a function of the state and gradient, i.e. to create a model for β as a function of the *local* $\mathbf{u}$ and $\nabla\mathbf{u}$. As the correction enters the equations at the residual level, propagation effects of convection-dominated systems can be modeled in this approach. For example, a discrepancy in the location of separation can be resolved through changes in the turbulent production at/before separation, where the boundary-layer profile can be controlled, instead of in the wake, where the largest state discrepancies occur

but where little control is possible.

The link between the state and the correction field can be made in the form of an analytical model [52], where the coefficients of the model can be fit to features of the state. Alternatively, the analytical model can be replaced altogether by a direct mapping from the state to the correction field, through an artificial neural network [53, 54]. This idea forms the basis of the field-inversion machine-learning (FIML) approach to corrected turbulence modeling [26].

FIML has seen success in making solutions of corrected turbulence models match unsteady or higher-fidelity data for a variety of cases [27, 29, 30, 32]. The generality of this approach is still in debate [25], and it is possible that FIML may never yield a general-purpose "new" turbulence model that performs better than existing analytical ones. In addition, the identification and selection of the most physically relevant local state features for use as inputs in to the neural network has a high degree of arbitrariness [30].

Fortunately, these possible shortcomings of FIML do not affect our present work, which does not aim to derive a general-purpose turbulence model. Our goal is to create a locally-valid corrected turbulence model that reproduces the behavior of the unsteady system at geometries and conditions that are close in parameter space. The model is retrained as the optimization progresses and more unsteady data become available. Furthermore, the fact that the model is specific to the problem simplifies the network input selection. Instead of choosing, perhaps arbitrarily, non-dimensional, physically meaningful features, we take a more systematic approach and use the entire state vector, its spatial gradient, and the wall distance as the network input.

B. Network Architecture and Training

The neural network maps local flowfield information to the scalar correction field. Figure 4 shows the structure of the network used in this work, which is a single-hidden-layer perceptron. The hidden layer, $\mathbf{x}_1$, contains n_1 neurons, between the input layer, i.e. the features, $\mathbf{x}_0$, and the output layer, which consists of the scalar correction factor, β. The

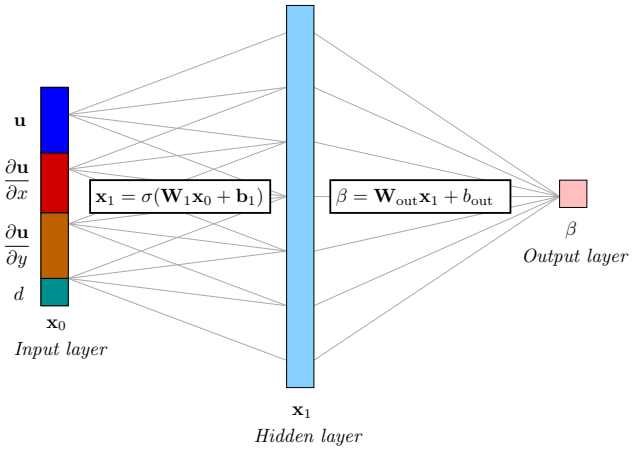


Fig. 4 Structure of the artificial neural networks used to predict the correction field.

map from the input to the hidden layer involves an entry-wise sigmoid activation function, $\sigma(x) = 1/(1+e^{-x})$, whereas no activation function is used for the output layer calculation. The parameters associated with the network consist of the weights and biases, $\mathbf{W}_i \in \mathbb{R}^{n_i \times n_{i-1}}$, $\mathbf{b}_i \in \mathbb{R}^{n_i}$, where n_i is the number of neurons in layer i.

The input into the network consists of the state, $\mathbf{u}$, its gradient, $\nabla\mathbf{u}$, and the wall distance, d, for a total of $(1+\dim)s+1$ neurons. After experimentation, the size of the hidden layer was set to $n_1 = 30$ for the two-dimensional problems considered here. The observed trade-off lies between over-fitting and high quadrature order requirements for large n_1 and a loss of accuracy for small n_1. A fairly wide range of values, $n_1 \in [16, 50]$, showed similar performance, so that the results are not overly sensitive to the precise value of n_1.

The parameters associated with the network consist of all of the weights and biases. The values of these parameters are determined using an optimization procedure, the adaptive moment (Adam) estimation algorithm in TensorFlow [55], that minimizes the mean squared error loss function between predicted and actual output-layer values. The actual output-layer values, i.e. values of the correction field, come from the field inversion result for one particular unsteady simulation. As $\beta(x)$ is a field, each simulation yields ostensibly an infinite amount of training data. However, the finite-dimensional representation of the state and $\beta(x)$ means that not all of these data are independent or informative for

training. In the present work, we sample the required fields (state, gradient, wall distance, correction) at the quadrature points of each element. Each inversion result then yields an amount of training data that depends on the mesh size and quadrature order.

The training data are broken into mini-batches of size 1000 for the optimizer, and the learning rate is set to .001. Prior to training, the weights and biases are initialized randomly from a unit normal distribution. Experiments with different learning rates, batch sizes, and initialization showed that the final results were not overly sensitive to these choices. 500,000 optimization iterations are taken in each training session for all of the presented results, although the mean-squared error typically stabilizes well before this number. Two to three orders of magnitude drop in the loss are usually observed.

C. Implementation and Example

Once a network is trained, it is implemented as a physics model in the turbulence source calculation of the flow simulation code. No changes to the inputs to this calculation are required, since the turbulent source already uses the state, its gradient, and the wall distance function. All parts of the source calculation must be differentiated with respect to the state and its gradient for the forward Newton solver and the discrete adjoint solver. This includes the neural-network correction field calculation, the linearization of which is calculated using back-propagation [53]. For a single-hidden layer network, the derivative of the scalar output with respect to the input vector is calculated as follows:

$$\beta = \mathbf{W}_{\text{out}}\sigma(\mathbf{W}_1\mathbf{x}_0 + \mathbf{b}_1) + b_{\text{out}} \quad \Rightarrow \quad \frac{\partial\beta}{\partial\mathbf{x}_0} = \mathbf{W}_2 \ \text{diag}(\boldsymbol{\sigma}_1') \ \mathbf{W}_1 \tag{22}$$

where $\boldsymbol{\sigma}_1'$ is the derivative of the activation function at the hidden layer values. For the sigmoid function, this derivative is $\sigma' = \sigma(1-\sigma)$, and hence its calculation only requires storing the hidden layer post-activation values. The chain rule calculation in Eqn. 22 extends systematically to multiple hidden layers. Due to the nonlinear nature of the network, higher-order accurate quadrature rules are used for the corrected RANS model. Presently, an constant order increment of 1 is added to the baseline $2p + 1$ order requirement in the code.

To demonstrate the ability of the neural-network model to produce an accurate correction field, we extend the example in Section IV.C to multiple airfoils of varying camber. For five airfoils of cambers 0 to 5% of the chord, unsteady simulations, field inversions, and neural network trainings are separately performed. The output of interest is the lift coefficient, and its sensitivity to camber changes.

Figure 5 shows the lift-coefficient results of the various simulations. The uncorrected RANS results show an increasing lift coefficient with camber, and adjoint-based gradients that match the trend from the multiple simulations. However, this trend disagrees with the time-averaged unsteady results, which show an initial decrease in the averaged lift coefficient for small cambers, followed by an increase. The result is not surprising, as the RANS model is not designed for such low Reynolds numbers.

Applying field inversion to each case and using the resulting correction field $\beta(\vec{x})$ in the turbulence model yields data that track the time-averaged results well. More importantly, the neural-network model for $\beta(\vec{x})$ also performs very well, as indicated by the "FIML" data points in Figure 5. These data correspond to simulations performed by using the neural network model and not the inverted $\beta(\vec{x})$ field, which was used only for training. Furthermore, when differentiated as part of the adjoint solver, the FIML result yields a reasonably-accurate PDE-constrained gradient of the lift coefficient with respect to the camber. The ability to accurately predict the lift coefficient value and its gradient with respect to shape parameters is important when using the FIML correction model in optimization.

Figure 6 shows the differences in the lift adjoint fields, the conservation of y momentum component, obtained from the uncorrected RANS and the FIML solutions. These are shown on the same scale, and the differences are large, particularly over the upper surface of the airfoil. The correction field, $\beta(\vec{x})$, is also shown, calculated using the neural network at the converged FIML state. This field matches well with the inversion result shown in Figure 3. Finally, the pressure distribution from the FIML state matches the averaged unsteady pressude distribution just as well as the field inversion result.

VI. Unsteady Optimization

The objective of the unsteady optimization problem is to determine the airfoil shape that minimizes a given time-averaged cost function, possibly subject to certain constraints. For example, we may be interested in minimizing the time-averaged drag coefficient subject to a fixed time-averaged lift coefficient and geometrical constraints such as

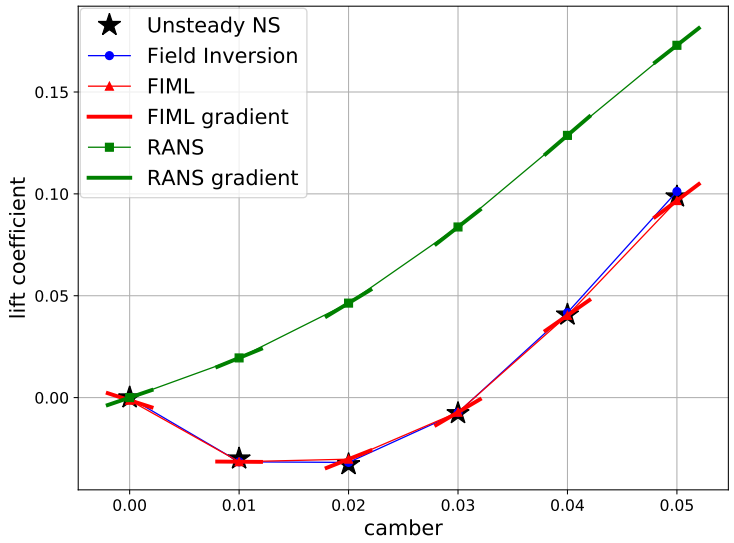
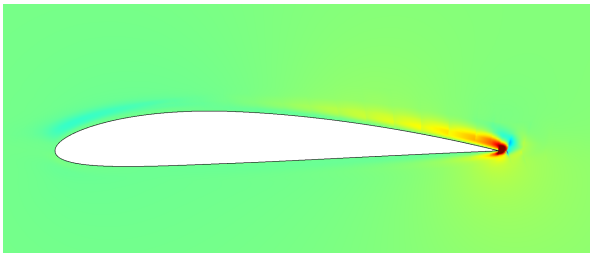
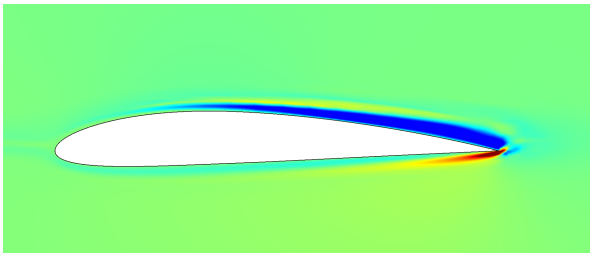


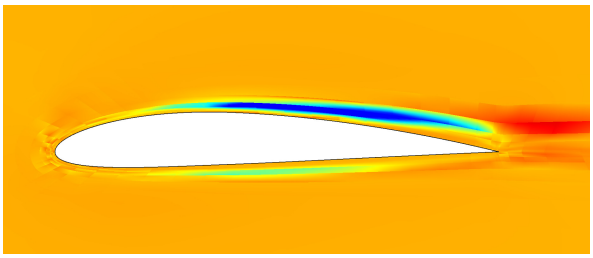
Fig. 5 Field-inversion and machine learning results for lift-coefficient calculations in unsteady flow around NACA X412 airfoils at $M = 0.2, Re = 10^4, \alpha = 0$.



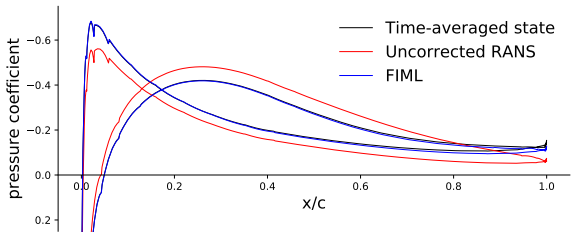
(a) Uncorrected RANS lift adjoint: y-momentum



(b) FIML lift adjoint: y-momentum



(c) Neural-network correction field, 0.3 – 1.3



(d) Surface pressure coefficient

Fig. 6 FIML and RANS contour plots for unsteady flow around a NACA 3412 airfoil at $M = 0.2, Re = 10^4, \alpha = 0$.

airfoil area and minimum thickness. Our chosen airfoil parametrization automatically enforces shape constraints of the latter form, and output constraints are handled via concurrent trimming.

In trimmed simulations, the trim conditions are enforced in steady mode prior to and then concurrently with the optimization, as described in Section III.A. In unsteady mode, trimming with one parameter is performed by averaging the trimming output (e.g. lift coefficient) over a prescribed time range and adjusting the trim parameter (e.g. angle of attack) at discrete intervals to achieve the target output. The sensitivities required for the parameter change are computed from a finite-difference of the previous two output and parameter values. The initial sensitivity comes from an adjoint-based derivative about the average state. A prescribed "burn" window, during which no averaging is performed, follows each parameter change and the initial time. Figure 7 summarizes this unsteady trimming procedure.

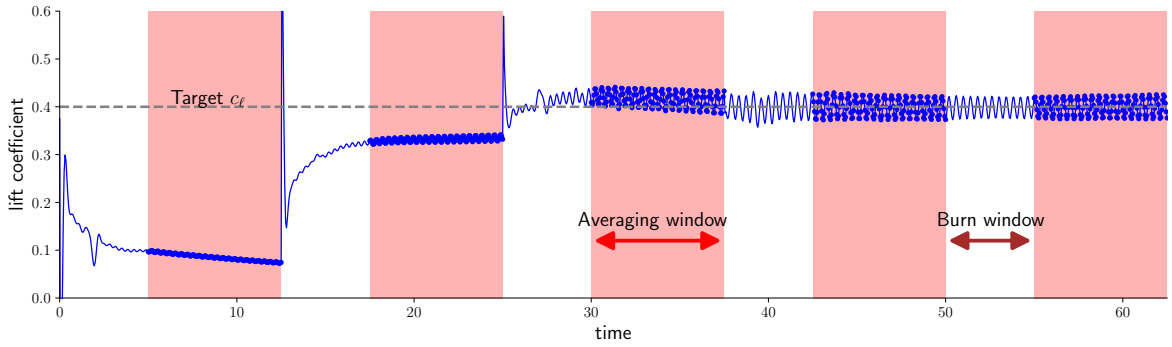


Fig. 7 Unsteady trimming of the lift coefficient using the angle of attack, taken from a simulation of the airfoil studied in Section VII.A.

The proposed unsteady optimization approach consists of the following steps:

1) Run the unsteady simulation, with trimming if needed, and save the average state.
2) Perform field inversion using the RANS model at the same geometry and free-stream conditions to determine the correction field, $\beta(x)$.
3) Train the neural network model to compute $\beta(x)$ as a function of the state, gradient, and wall distance.
4) Optimize the geometry in steady mode using the correction model, i.e. FIML.
5) Return to step 1.

Each cycle through these steps constitutes an unsteady optimization iteration. The initial airfoil shape for the first unsteady simulation could be based on default parameters or it could be the result of a steady-state optimization. The latter is generally more efficient, although we present both approaches in the results.

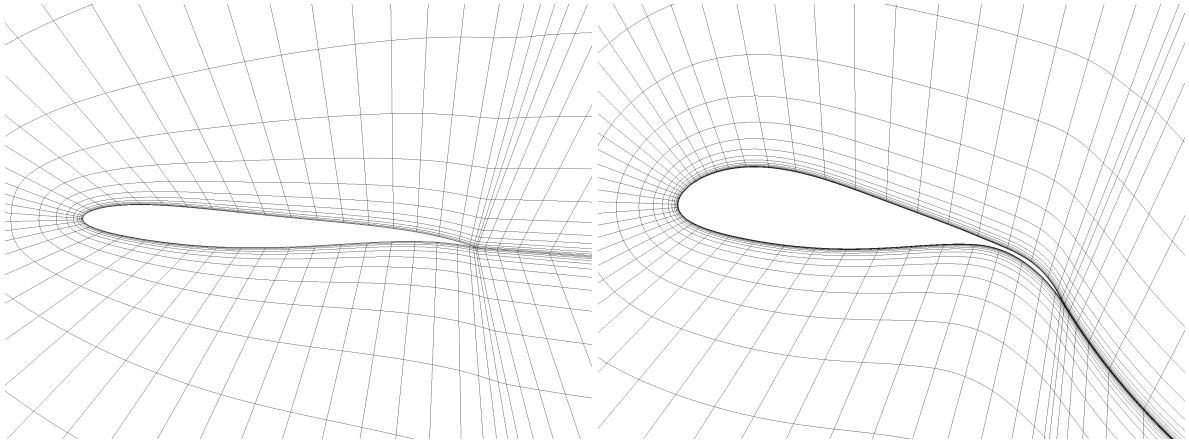
VII. Results

This section presents a sequence of three numerical results that demonstrate the unsteady optimization method introduced in this work. The computational meshes in each case consist of $q = 3$ curved quadrilaterals generated by a script according to the specified shape parameters. Anisotropic elements resolve the boundary layer through a non-uniform airfoil-normal spacing chosen a priori based on the Reynolds number. Figure 8 shows the meshes, at the optimized geometries, for the three cases considered. The angle of attack is imposed through a static arbitrary Lagrangian-Eulerian pitch deformation centered at the airfoil mid-chord and blended to zero $20c$ away from the center.

A. Fixed-Lift Drag Minimization, $Re = 10^4$

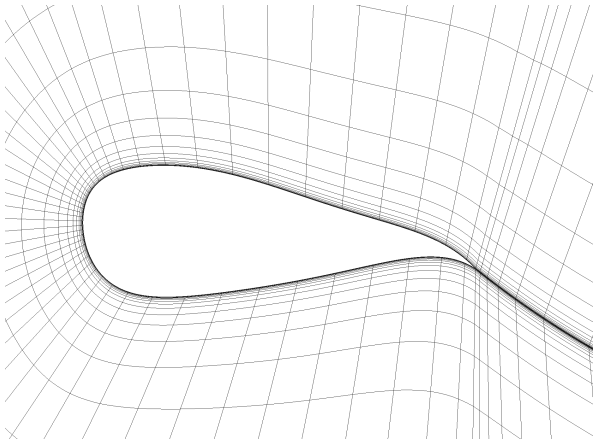
As a first example, we consider a drag minimization problem for an airfoil at a prescribed lift coefficient of $c_\ell = 0.4$, and $M = 0.2$, $Re = 10^4$. No turbulence model is used in the unsteady simulations at this low Reynolds number. The area of the airfoil is constrained to $A = .06c^2$ via Eqn. 17. Three camber parameters, c_0, c_1, c_2 in Eqn. 14, and two thickness parameters, t_1, t_2 in Eqn. 15, dictate the shape of the airfoil. All five of these parameters are set to zero to obtain the initial shape.

In this case, we first trim the initial airfoil in unsteady mode to determine the lift coefficient that attains the prescribed average lift coefficient. Figure 9(a) shows the lift coefficient time history for a portion of the trimmed condition, along



(a) $Re = 10^4$, $c_\ell = 0.4$: 1978 elements

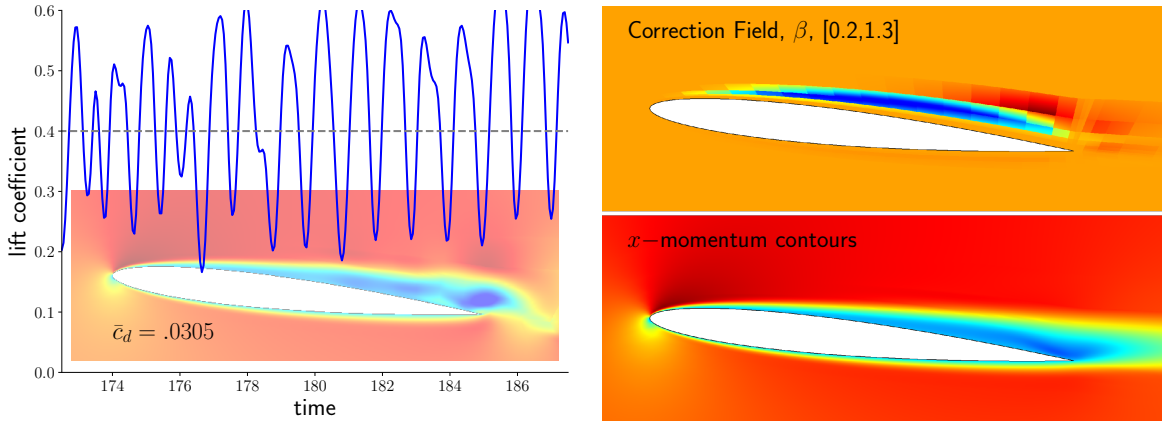
(b) $Re = 10^6$, $c_\ell = 2.4$: 2816 elements



(c) $Re = 10^6$, $\max(L/D)$: 2816 elements

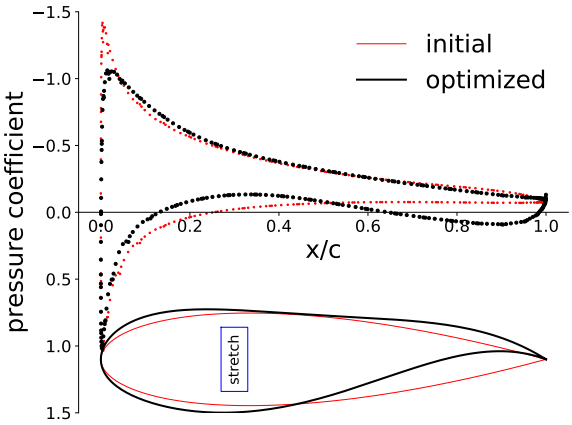
Fig. 8 Meshes used for the three results cases, shown at the optimized geometries.

with a snapshot of the x-momentum contour field. The initial airfoil exhibits large oscillations in the lift coefficient, in the approximate range $c_\ell \in [0.2, 0.6]$ and frequency of $1c/U_\infty$, where U_∞ is the freestream speed.



(a) Unsteady solution

(b) Field inversion



(c) FIML Optimization

Fig. 9 First FIML optimization iteration for minimizing drag on an airfoil at $M = 0.2, Re = 10^4, c_\ell = 0.4$.

The oscillatory flowfield and outputs prevent the application of standard unsteady adjoint techniques for computing gradients [51, 56, 57]. Figure 10 shows the backward-in-time growth of the adjoint solution norm for a time-integral lift output, along with snapshots of the instantaneous adjoint. In addition to this instability, a full unsteady adjoint solution is also expensive in storage an computational overhead of the reverse time-marching solution, rendering it impractical for unsteady turbulent flows. We stress that the present approach does not require unsteady adjoints, as it creates a tailored steady-state model for the time-averaged solution and optimizes using this model.

The time-average state and outputs for the initial airfoil were obtained using $20c/U_\infty$ time units of the trimmed unsteady simulation. The average values were used as targets in the field inversion process, as described in Section IV. In particular, 50 steepest-descent iterations were taken to determine the $\beta(\vec{x})$ field that minimized the error in the airfoil surface stress distribution. Figure 9(b) show the resulting correction field and x-momentum contours from the corrected RANS simulation. Note that the correction field calls for a general decrease of turbulent production at the edge of the upper-surface boundary layer, which leads to earlier separation and higher drag compared to an uncorrected-RANS simulation.

Data from this correction field were used to train a neural network, with one hidden layer of 30 neurons, to predict β from the local state, gradient, and wall-distance function. The RANS equations together with the network-based correction field constitute the FIML model, which is used in a steady shape optimization. Figure 9(c) shows the result of 50 BFGS optimization iterations using this first FIML model. The optimized shape exhibits a reduced suction peak, a

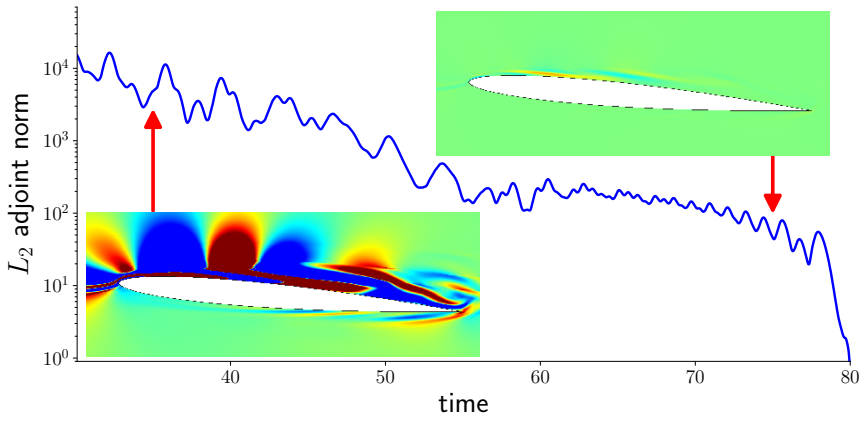


Fig. 10 Instability in the unsteady lift-integral adjoint for the initial airfoil in the constrained-lift drag minimization at $M = 0.2, Re = 10^4, c_\ell = 0.4$. Field plots are of the conservation of x momentum component of the adjoint. Time is measured in units if airfoil chord divided by the freestream speed.

higher aft-loading redistribution of the lift, and a shift of area from the aft the front of the airfoil.

The combination of unsteady simulation, field inversion, and corrected steady optimization constitutes one *FIML optimization iteration*. This process is repeated with the optimized airfoil, and Figure 11 shows the results of the second FIML optimization iteration, which parallel those of Figure 9.

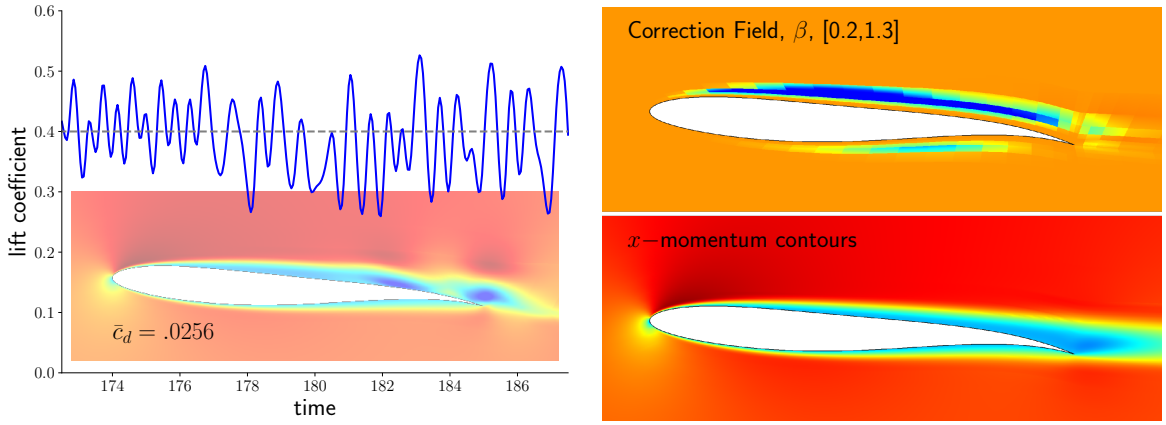
The flowfield remains unsteady for the optimized airfoil, but the unsteady trimming time history in Figure 11(a) shows a reduced amplitude of the lift coefficient variation, by almost a factor of two compared to Figure 9(a). Field inversion was then performed using the time-averaged surface stress distribution as a target and the same parameters as in the first iteration. Figure 11(b) shows the resulting correction field and x-momentum contours. The dominant feature of the correction field is again a suppression of turbulent production on the upper surface. A similar but lower-magnitude suppression now also appears on the lower surface due to the adverse pressure gradient brought about by the increased aft camber.

A new neural network is constructed from the second correction field, and the resulting FIML model is used to re-optimize the airfoil shape. Figure 11(c) shows the shape and pressure distribution of the second FIML optimization iteration. The changes in the shape compared to the first FIML optimization are not large: a slight flattening of the upper surface and reduction in overall camber, except for the aft portion.

Two additional FIML optimizations were performed, each with a separate unsteady simulation, inversion, and steady optimization. Figure 12 shows the final FIML-optimized airfoil shape, along with a comparison to the uncorrected-RANS optimization result. The latter shape, referred to simply as the RANS airfoil, exhibits a rounder upper surface and a more aggressive thickness decrease along the chord, which leads to a larger-magnitude adverse pressure gradient. An uncorrected RANS analysis of this airfoil yields the Mach number contours shown in Figure 12(b): the flow remains attached on the upper and lower surfaces. However, an unsteady analysis without the RANS model results in a different picture. Figure 12(c) shows the Mach number from the time-averaged state, and the wake is much larger due to separated flow on the upper surface. In contrast, the FIML-optimized airfoil yields a time-averaged state with a smaller wake due to, on average, later separation enabled by a less adverse pressure gradient.

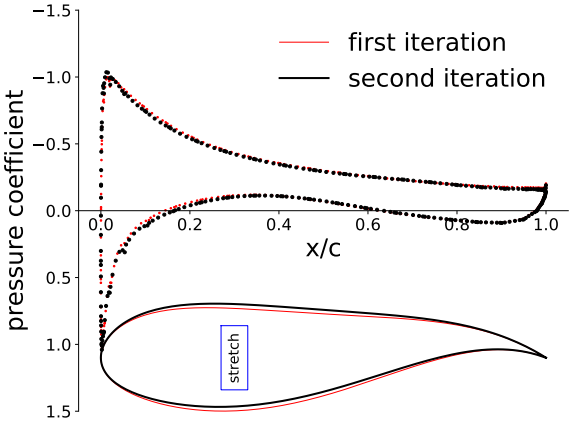
Table 1 summarizes the optimization results in terms of the average drag coefficient from a trimmed unsteady simulation of the various airfoils. RANS-alone optimization does result in a drag reduction by almost 50 counts compared to the chosen initial airfoil. This is similar to the result from the first iteration of the FIML optimization. However, subsequent FIML optimizations further reduce the average drag by over 40 drag counts.

To verify the importance of the correction model on the optimal shape, the final FIML-optimized airfoil was used as the initial airfoil in an uncorrected-RANS optimization. The result of this optimization was the same RANS-alone optimized result obtained from the original initial airfoil, shown in Figure 12. This indicates that the FIML correction-field model is responsible for the additional over 40 counts of drag reduction compared to an uncorrected RANS optimization.



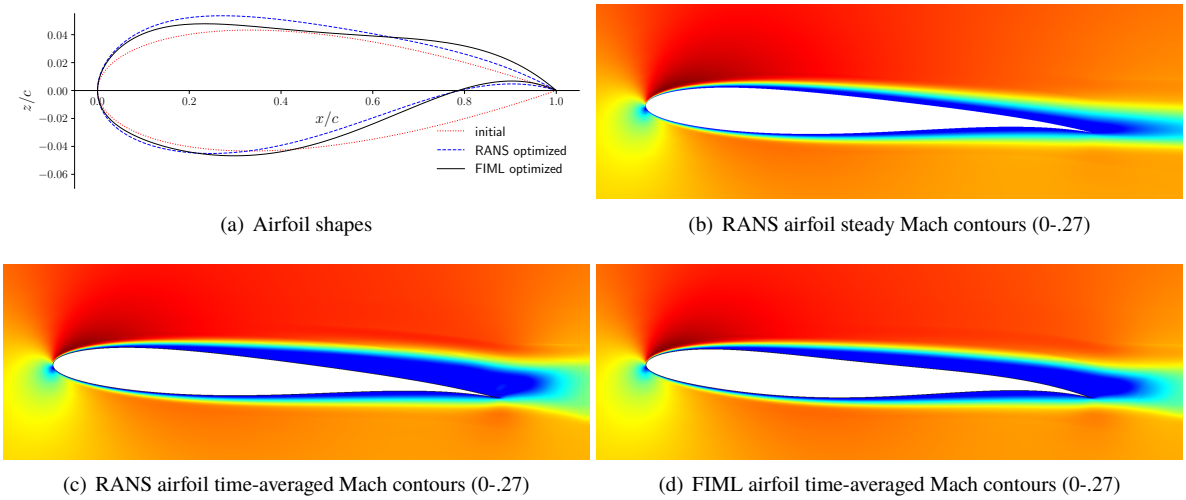
(a) Unsteady solution

(b) Field inversion



(c) FIML Optimization

Fig. 11 Second FIML optimization iteration for minimizing drag on an airfoil at $M = 0.2, Re = 10^4, c_\ell = 0.4$.



(a) Airfoil shapes

(b) RANS airfoil steady Mach contours (0-.27)

(c) RANS airfoil time-averaged Mach contours (0-.27)

(d) FIML airfoil time-averaged Mach contours (0-.27)

Fig. 12 Comparison of RANS and FIML optimized airfoil shapes and time-averaged Mach number contours at $M = 0.2, Re = 10^4, c_\ell = 0.4$.

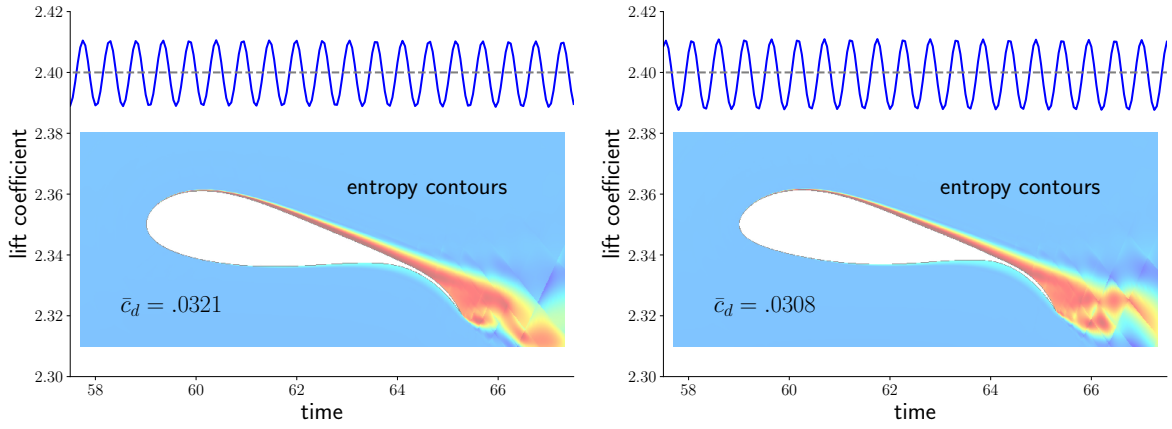
Table 1 Time-averaged drag coefficients of initial and optimized airfoils at $M = 0.2, Re = 10^4, c_\ell = 0.4$.

Airfoil	Average drag coefficient
Initial airfoil shape	0.0305
RANS-optimization	0.0259
FIML-optimization, 1st iteration	0.0256
FIML-optimization, 2nd iteration	0.0241
FIML-optimization, 3rd iteration	0.0215
FIML-optimization, 4th iteration	0.0212

B. Fixed-Lift Drag Minimization, $Re = 10^6$

We next consider a drag minimization problem at a higher lift coefficient, $c_\ell = 2.4$ and Reynolds number, $Re = 10^6$. The free-stream Mach number remains $M = 0.2$, but the airfoil is thicker, with area constrained to $A = 0.1c^2$. Four camber parameters, c_0, c_1, c_2, c_4 in Eqn. 14, and three thickness parameters, t_1, t_2, t_3 in Eqn. 15, dictate the shape of the airfoil.

In this case, we first optimize the airfoil using the standard, uncorrected RANS equations. The resulting shape, termed the RANS airfoil, is then trimmed in unsteady mode, using the simple DES model described in Section II.A, with $d_{\max} = .02c$. The time-averaged state and output from the trimmed unsteady simulation provide the target data for field inversion, machine learning, and the follow-up FIML optimization. We refer to the resulting airfoil as the FIML airfoil, and we compare its performance to that of the RANS airfoil.



(a) RANS airfoil (b) FIML airfoil

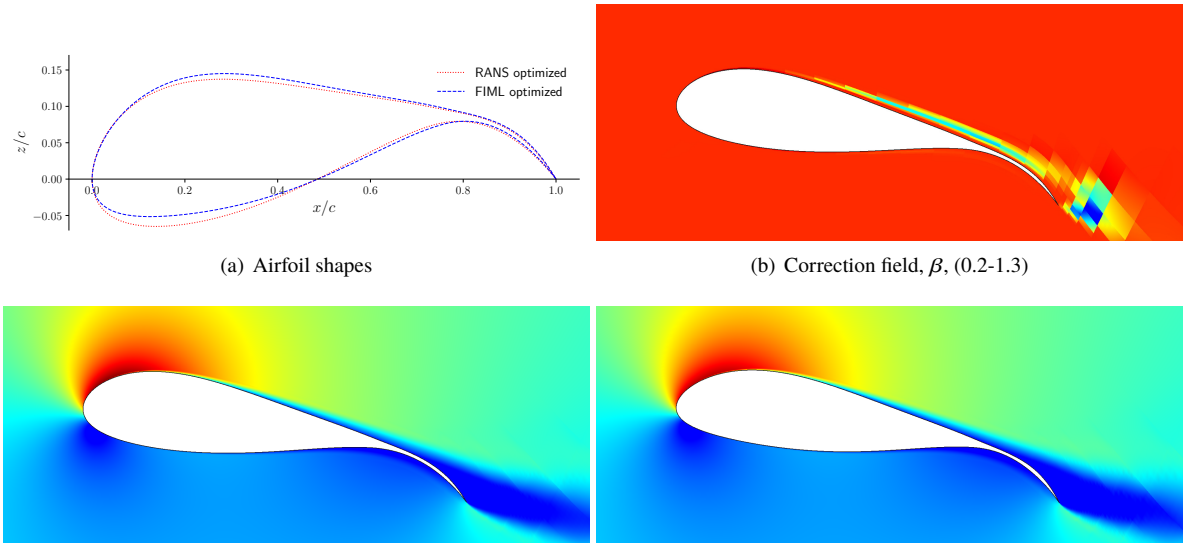
Fig. 13 Unsteady analyses of RANS and FIML airfoils optimized for minimizing drag at $M = 0.2, Re = 10^6, c_\ell = 2.4$.

Figure 13 shows the unsteady lift coefficient time histories, after trimming, of both the RANS and the FIML airfoils. These are similar and exhibit small variations about the target mean. The flow develops unsteadiness in the wake of the upper surface, as shown by the accompanying entropy contours.

The optimized airfoil shapes, shown more clearly in the subsequent figures, exhibit a couple of interesting features. First, the aft portion of the airfoil is very thin. The thickness constraint is active, as most of the required airfoil area is placed towards the front of the airfoil. The aft portion is also highly cambered, and thus responsible for much of the lift generation as well as the unsteadiness due to separation. However, as this is a small region, the resulting unsteadiness does not dominate the flowfield the flow remains attached over the majority of the airfoil.

Figure 14(a) compares the RANS and FIML airfoil shapes. We see that the shapes are quite similar, with the FIML airfoil redistributing some of the thickness from the front of the airfoil over a larger portion of the chord. The FIML airfoil also has a slightly sharper leading edge. The correction field obtained after field inversion, shown in Figure 14(b),

shows a thin region of suppression of turbulence production over the upper surface, which is responsible for boundary layer thickening and an earlier point of separation. The correction is largely inactive over the lower surface but flares up in the wake, shortly behind the trailing edge, in the form of a confined region of suppression of turbulent production. This suppression reduces mixing in the shear layer between the lower and upper surface flows.



(a) Airfoil shapes

(b) Correction field, β, (0.2-1.3)

(c) RANS airfoil steady Mach contours (0-.5)

(d) FIML airfoil time-averaged Mach contours (0-.5)

Fig. 14 Comparison of RANS and FIML airfoil shapes, the correction field, and Mach number contours for drag minimization at $M = 0.2, Re = 10^6, c_\ell = 2.4$.

Finally, Figure 14 also shows the Mach number contours from both the steady and the time-averaged unsteady flowfields of the RANS and FIML airfoils, respectively. The high Reynolds number of this flow makes the differences difficult to discern, but both the upper-surface boundary layer and the wake are thicker in the time-averaged FIML-airfoil case than in the steady RANS airfoil case. These differences impact the outputs: the steady RANS analysis predicts a drag coefficient that is 46 counts lower than the time-averaged result.

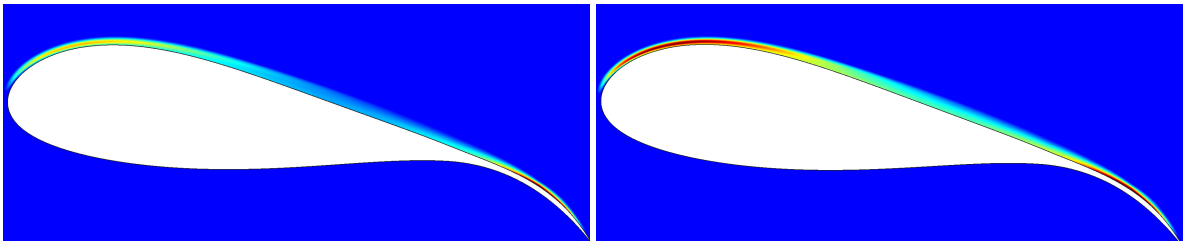
Table 2 summarizes the drag coefficient results for the two airfoils. To recap, the RANS airfoil is a result of a steady optimization using the standard RANS equations. It is analyzed both in steady and unsteady, DES, modes. The first two lines of Table 2 show the differences in the trimmed-state drag coefficients between these two analyses of the RANS airfoil: 45 counts higher for the unsteady analysis. The time-averaged unsteady analysis of the RANS airfoil provides the target data for the field inversion and training of the FIML model, an optimization of which yields the FIML airfoil. The parametrization remains the same: seven shape parameter and the angle of attack as the trim parameter. The last line of Table 2 shows the unsteady-analysis drag coefficient result for the FIML airfoil: a reduction of 13 drag counts relative to the RANS airfoil. As in the previous case, the shape difference between the two airfoils was tested for persistence: an uncorrected RANS optimization starting with the FIML airfoil reverts back to the RANS airfoil.

Table 2 Drag coefficients of optimized airfoils at $M = 0.2, Re = 10^6, c_\ell = 2.4$.

Airfoil	Drag coefficient
RANS airfoil, steady analysis	0.0275
RANS airfoil, unsteady analysis	0.0321
FIML airfoil, unsteady analysis	0.0308

Although the primal flowfields of the RANS and FIML airfoils are similar, differences in the adjoint solutions are more pronounced. Figure 15 compares the drag adjoints obtained from the uncorrected RANS equations and the RANS equations with the FIML correction, referred to as the FIML equations. Both fields are shown on the same shape, the RANS airfoil, and the adjoint component shown corresponds to the conservation of x-momentum equation. We see that

for the FIML equations, the drag is more sensitive to residual perturbations along the upper surface, close to the airfoil, particularly in the acceleration region over the front of the airfoil and the high-camber aft portion. These differences in the adjoint arise from differences in both the primal state and the residual Jacobian matrix. The latter is caused by the linearization of the neural-network model for the correction field, which affects the matrix of the adjoint linear system.



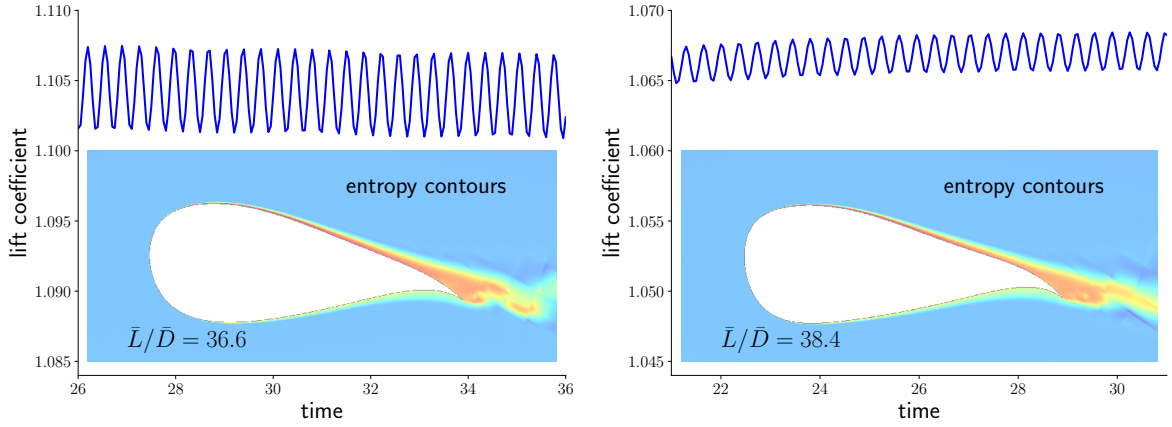
(a) RANS-equation adjoint (b) FIML-equation adjoint

Fig. 15 Field plots of the conservation of x-momentum component of the drag adjoint for the RANS-optimized airfoil, using both the RANS equations and the corrected, FIML equations at $M = 0.2, Re = 10^6, c_\ell = 2.4$.

C. Lift-to-Drag Ratio Maximization, $Re = 10^6$

As a final example, we consider the maximization of the lift-to-drag ratio of an airfoil at free-stream $M = 0.2$ and $Re = 10^6$. The airfoil is thicker than in the previous examples, with an area constraint of $A = 0.2c^2$. The design parameters consist of the same seven shape parameters as in the previous example, Section VII.B, augmented by the angle of attack. Note that we do not have trim parameters in this unconstrained maximization problem.

As in the previous example, we first optimize the airfoil using the standard, uncorrected RANS equations, and this optimization yields the RANS airfoil. An unsteady analysis of the RANS airfoil, shown in Figure 16(a), is then performed using the simple DES model described in Section II.A, with $d_{\max} = .02c$. The flowfield is unsteady, as evidenced by the lift coefficient output history and the entropy contour snapshot. The variation of the lift coefficient around the mean is relatively high in frequency, compared to the lower Reynolds number example, but small in amplitude.



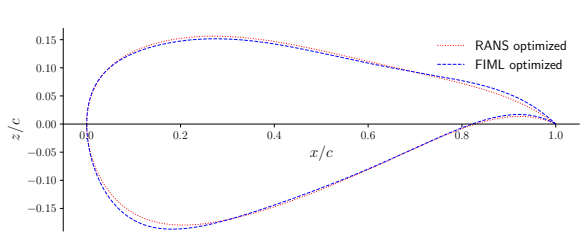
(a) RANS airfoil (b) FIML airfoil

Fig. 16 Unsteady analyses of RANS and FIML airfoils optimized for maximizing the lift-to-drag ratio at $M = 0.2, Re = 10^6$.

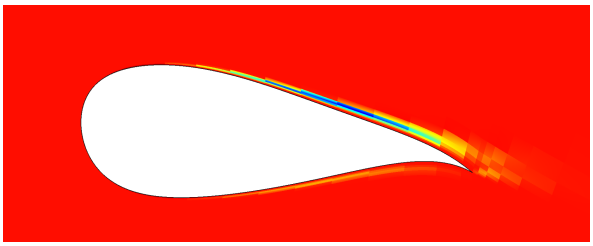
Time-averaged state and output information from the unsteady analysis of the RANS airfoil provides the target data for field inversion and machine learning. The resulting FIML model is used to re-optimize the airfoil with unsteady information. The optimization parallels the steady RANS case, and it is in fact cheaper in this case, as the RANS airfoil provides a good initial guess for the optimal shape. Figure 16(b) provides a snapshot of the unsteady analysis of the

resulting FIML airfoil. Although the vertical axis range changes due to a different average lift coefficient, the scale remains the same. We see that the amplitude of the lift coefficient variation decreases for the FIML airfoil, and the average lift-to-drag ratio improves from 36.6 to 38.4.

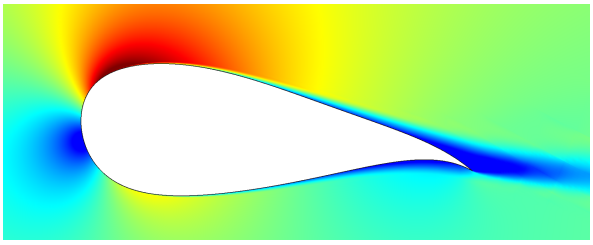
Figure 17 compares the RANS and FIML airfoils in more detail. Starting with the shape, shown in Figure 17(a), we see that again, the RANS and FIML airfoils are generally similar. Both exhibit an area concentration towards the front of the airfoil, and a thinner, more cambered, trailing edge. Due to the overall area constraint of the airfoil, no parts of the airfoil are up against the thickness constraint. The FIML airfoil shows slightly increased thickness at the leading edge, a less aggressive upper surface curvature, and increased camber at the trailing edge. These differences, although subtle, nevertheless lead to improved unsteady performance, as measured by the lift-to-drag ratio.



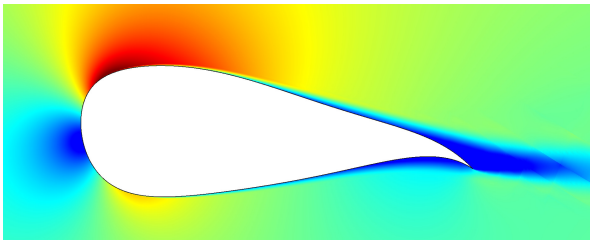
(a) Airfoil shapes



(b) Correction field, β, (0.5-1.1)



(c) RANS airfoil steady Mach contours (0-.4)



(d) FIML airfoil time-averaged Mach contours (0-.4)

Fig. 17 Comparison of RANS and FIML airfoil shapes and time-averaged Mach number contours for lift-to-drag ratio maximization at $M = 0.2, Re = 10^6$.

Figure 17(b) shows the correction field obtained from the field inversion of the unsteady analysis of the RANS airfoil. We see that the correction field consists primarily of a thin layer of turbulent production suppression over the upper surface, which is responsible for less mixing, a weaker boundary layer, and earlier separation of the flow over the upper surface. This correction makes the FIML solution more closely match the unsteady solution. As the unsteady flow separates earlier, the FIML optimization does not try to keep it attached over the upper-surface of the trailing-edge "flap" portion of the airfoil. Instead, the FIML optimization accepts the separation and associated drag increase, and it increases the camber there so that more lift can be generated there at a lower angle of attack by the redirection of the lower-surface flow. These features are subtly evident in the Mach number contours shown in Figure 17(c) and 17(d).

Finally, Table 3 summarizes the unsteady, time-averaged outputs computed for both the RANS and the FIML airfoils. We see that the RANS airfoil has a larger lift coefficient but also a larger drag coefficient compared to the FIML airfoil. This results in a lift-to-drag ratio that is lower, in the time-averaged unsteady analysis, compared to that of the FIML airfoil. As in the previous cases, the differences are persistent, in that re-running the uncorrected RANS optimization starting from the FIML airfoil reverts to the RANS airfoil.

Table 3 Time-averaged outputs for lift-to-drag ratio maximization at $M = 0.2, Re = 10^6$.

Airfoil	Average lift coefficient	Average drag coefficient	$\bar{L}/\bar{D}$
RANS airfoil	1.104	.0301	36.6
FIML airfoil	1.067	.0278	38.4

VIII. Conclusions

This paper introduces a gradient-based approach for optimizing shapes of airfoils in unsteady turbulent flows. The most challenging aspect of this problem is the calculation of the gradient, which is the sensitivity of the output with respect to the shape parameters. Whereas adjoint-based methods enable efficient gradient calculations, their application to unsteady turbulent flows is hampered by the fact that for problems exhibiting chaotic or limit-cycle oscillations, a direct unsteady adjoint solution is unstable. Furthermore, unsteady adjoint solutions are inherently expensive, becoming impractical for large simulations in which only a few forward primal solutions can be afforded.

The gradient calculation proposed in this work does not rely on an unsteady adjoint. Instead, a steady-state model is first created from the unsteady simulation data, and the adjoint method is applied to the steady-state model. The steady-state model is based on the Reynolds-averaged Navier-Stokes equations, to which a correction is applied via a multiplicative factor on the production term. The correction factor is a field quantity that is computed from the unsteady simulation data by solving an inverse problem. The unsteady simulation provides the target data for this inversion.

Following the field inversion, a neural network model is trained to produce the correction factor from local flow data. The single corrected RANS solution provides a large quantity of training data, as the correction factor is measured at the quadrature points of each element. The local data consist simply of the state, its spatial gradient, and the wall distance. No effort is made to form non-dimensional inputs, as the network does not need to be generalizable across different flow regimes or vastly-different geometries. Instead, the network is used only for a single steady-state optimization. The network must be linearized for accurate adjoint calculations. After shape and trim parameter changes, a new unsteady run follows, and a new network is trained. Thus, the goal of the machine learning as used in this work is not to create a new turbulence model, but to enable the calculation of an adjoint for an unsteady flow.

The discretization used in this work is a discontinuous Galerkin (DG) finite-element method with implicit time stepping. DG allows for a high-order solution representation that can be more efficient at resolving the smooth unsteady flows considered here, with features at multiple length and time scales. However, the unsteady optimization approach is not tied to DG and could be used with other discretizations, such as finite-volume or finite-difference methods. Similarly, the method is not tied to the unsteady model, which here is a very simple approximation to DES. Indeed, as the method is not intrusive into the unsteady simulation, independent "black-box" unsteady solvers could be used, with various levels of fidelity, including DES or LES. Only average flowfield statistics are required for the inversion and training. Finally, the approach is not tied to the specific steady-state optimization method, as long as it is gradient-based. This work uses the BFGS method with concurrent trimming, and a polynomial-based parametrization of the airfoil camber and thickness. Other, more sophisticated optimization approaches could also be used.

The ideal application of the presented unsteady optimization approach is to turbulent flows that are unsteady but reasonably well-approximated by the baseline steady turbulence model, here RANS-SA. For airfoils, this usually means high-lift conditions or large thickness. An airfoil at on-design cruise conditions would likely be better optimized using the RANS model alone. At the other end of the spectrum, a flow with massive separation may be too far from the range of RANS applicability for the correction to be sufficient. As such, the results presented here focus on thick airfoils at high-lift. These include drag minimization at high lift and maximization of the lift-to-drag ratio of a thick airfoil. In both of these cases, the unsteady optimization yields improved designs: an average drag reduction of over 4% and a lift-to-drag ratio improvement of almost 5%, respectively, over a standard RANS optimization. These are modest gains as RANS is already a decent model for these unsteady flows.

More interesting is the low-Reynolds number result, fixed-lift drag minimization at $Re = 10^4$. In this case, the RANS model by itself is not accurate, as it predicts attached flow whereas the averaged unsteady flowfield has a high propensity for separation. With a correction, primarily suppression of turbulent production, the FIML model is able to very well approximate the averaged unsteady flowfield and outputs. Moreover, the gradients computed from the FIML model also match reasonably well with the unsteady flowfield gradients. As a result, optimization with the FIML model leads to a design that has an 18% lower averaged drag coefficient compared to the RANS optimization, when both are analyzed using the unsteady Navier-Stokes equations. This result shows that the unsteady optimization approach can be applied to cases in which the original RANS model would not be deemed appropriate. Pushing the envelope of the FIML/RANS combination, e.g. to stalled conditions or bluff-bodies, is the subject of current work.

References

[1] Karbasian, H. R., and Vermeire, B. C., "Gradient-free aerodynamic shape optimization using large eddy simulation," *Computers and Fluids*, Vol. 232, 2022, p. 105185. https://doi.org/https://doi.org/10.1016/j.compfluid.2021.105185.

[2] Pironneau, O., "On optimum design in fluid mechanics," *Journal of Fluid Mechanics*, Vol. 64, 1974, pp. 97–110. https://doi.org/https://doi.org/10.1017/S0022112074002023.

[3] Nadarajah, S. K., and Jameson, A., "A Comparison Of The Continuous And Discrete Adjoint Approach To Automatic Aerodynamic Optimization," AIAA Paper 2000-0667, 2000. https://doi.org/https://doi.org/10.2514/6.2000-667.

[4] Martins, J. R. R. A., and Lambe, A. B., "Multidisciplinary design optimization: a survey of architectures," *AIAA Journal*, Vol. 51, No. 9, 2013, pp. 2049–2075. https://doi.org/https://doi.org/10.2514/1.J051895.

[5] Economon, T. D., Palacios, F., and Alonso, J. J., "Unsteady Continuous Adjoint Approach for Aerodynamic Design on Dynamic Meshes," *AIAA Journal*, Vol. 53, No. 9, 2015, pp. 2437–2453. https://doi.org/https://doi.org/10.2514/1.J053763.

[6] Chen, G., and Fidkowski, K. J., "Discretization error control for constrained aerodynamic shape optimization," *Journal of Computational Physics*, Vol. 387, 2019, pp. 163–185. https://doi.org/10.1016/j.jcp.2019.02.038.

[7] Johnson, C., "On Computability and Error Control in CFD," *International Journal for Numerical Methods in Fluids*, Vol. 20, 1995, pp. 777–788. https://doi.org/https://doi.org/10.1002/fld.1650200806.

[8] Wang, Q., "Forward and adjoint sensitivity computation of chaotic dynamical systems," *Journal of Computational Physics*, Vol. 235, 2013, pp. 1–13. https://doi.org/https://doi.org/10.1016/j.jcp.2012.09.007.

[9] Wang, Q., Hu, R., and Blonigan, P., "Least Squares Shadowing sensitivity analysis of chaotic limit cycle oscillations," *Journal of Computational Physics*, Vol. 267, 2014, pp. 210–224. https://doi.org/https://doi.org/10.1016/j.jcp.2014.03.002.

[10] Wang, Q., "Convergence of the least squares shadowing method for computing derivative of ergodic averages," *SIAM Journal on Numerical Analysis*, Vol. 52, No. 1, 2014, pp. 156–170. https://doi.org/https://doi.org/10.1137/130917065.

[11] Blonigan, P. J., Gomez, S. A., and Wang, Q., "Least Squares Shadowing for Sensitivity Analysis of Turbulent Fluid Flows," AIAA Paper 2014–1426, 2014. https://doi.org/https://doi.org/10.2514/6.2014-1426.

[12] Ni, A., and Wang, Q., "Sensitivity Analysis on Chaotic Dynamical Systems by Non-Intrusive Least Squares Shadowing (NILSS)," *Journal of Computational Physics*, Vol. 347, 2017, pp. 56–77. https://doi.org/10.1016/j.jcp.2017.06.033.

[13] Ni, A., Wang, Q., Fernández, P., and Talnikar, C., "Sensitivity analysis on chaotic dynamical systems by Finite Difference Non-Intrusive Least Squares Shadowing (FD-NILSS)," *Journal of Computational Physics*, Vol. 394, 2019, pp. 615–631. https://doi.org/doi.org/10.1016/j.jcp.2019.06.004.

[14] Griewank, A., and Walther, A., "Revolve: An Implementation of Checkpointing for the Reverse or Adjoint Mode of Computational Differentiation," *ACM Transactions on Mathematical Software*, Vol. 26, No. 1, 2000, pp. 19–45. https://doi.org/https://doi.org/10.1145/347837.347846.

[15] Meidner, D., and Vexler, B., "Adaptive Space-Time Finite Element Methods for Parabolic Optimization Problems," *SIAM Journal on Control Optimization*, Vol. 46, No. 1, 2007, pp. 116–142. https://doi.org/https://doi.org/10.1137/060648994.

[16] Mani, K., and Mavriplis, D. J., "Error Estimation and Adaptation for Functional Outputs in Time-Dependent Flow Problems," *Journal of Computational Physics*, Vol. 229, 2010, pp. 415–440. https://doi.org/https://doi.org/10.1016/j.jcp.2009.09.034.

[17] Fidkowski, K. J., and Luo, Y., "Output-based Space-Time Mesh Adaptation for the Compressible Navier-Stokes Equations," *Journal of Computational Physics*, Vol. 230, 2011, pp. 5753–5773. https://doi.org/10.1016/j.jcp.2011.03.059.

[18] Fidkowski, K. J., "An Output-Based Dynamic Order Refinement Strategy for Unsteady Aerodynamics," AIAA Paper 2012-77, 2012. https://doi.org/10.2514/6.2012-77.

[19] Kast, S. M., and Fidkowski, K. J., "Output-based Mesh Adaptation for High Order Navier-Stokes Simulations on Deformable Domains," *Journal of Computational Physics*, Vol. 252, No. 1, 2013, pp. 468–494. https://doi.org/10.1016/j.jcp.2013.06.007.

[20] Fidkowski, K. J., "Output-Based Space-Time Mesh Optimization for Unsteady Flows Using Continuous-in-Time Adjoints," *Journal of Computational Physics*, Vol. 341, No. 15, 2017, pp. 258–277. https://doi.org/10.1016/j.jcp.2017.04.005.

[21] Ojha, V., Fidkowski, K. J., and Cesnik, C. E. S., "Adaptive Mesh Refinement for Fluid-Structure Interaction Simulations," AIAA Paper 2021–0731, 2021. https://doi.org/10.2514/6.2021-0731.

[22] Belme, A., Dervieux, A., and Alauzet, F., "Time accurate anisotropic goal-oriented mesh adaptation for unsteady flows," *Journal of Computational Physics*, Vol. 231, No. 19, 2012, pp. 6323–6348. https://doi.org/10.1016/j.jcp.2012.05.003.

[23] Alauzet, F., Loseille, A., and Olivier, G., "Time accurate anisotropic goal-oriented mesh adaptation for unsteady flows," *Journal of Computational Physics*, Vol. 373, No. 15, 2018, pp. 28–63. https://doi.org/10.1016/j.jcp.2018.06.043.

[24] Larsson, J., and Wang, Q., "The prospect of using large eddy and detached eddy simulations in engineering design, and the research required to get there," *Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, Vol. 372, No. 2022, 2014, p. 20130329. https://doi.org/https://doi.org/10.1098/rsta.2013.0329.

[25] Rumsey, C., Coleman, G., and L.Wang, "In Search of Data-Driven Improvements to RANS Models Applied to Separated Flows," AIAA Paper 2022–0937, 2022. https://doi.org/10.2514/6.2022-0937.

[26] Parish, E. J., and Duraisamy, K., "A paradigm for data-driven predictive modeling using field inversion and machine learning," *Journal of Computational Physics*, Vol. 305, 2016, pp. 758–774. https://doi.org/10.1016/j.jcp.2015.11.012, URL https://www.sciencedirect.com/science/article/pii/S0021999115007524.

[27] Singh, A. P., Medida, S., and Duraisamy, K., "Machine-learning-augmented predictive modeling of turbulent separated flows over airfoils," *AIAA Journal*, Vol. 55, No. 7, 2017, pp. 2215–2227. https://doi.org/10.2514/1.j055595.

[28] Singh, A. P., Pan, S., and Duraisamy, K., "Characterizing and Improving Predictive Accuracy in Shock-Turbulent Boundary Layer Interactions Using Data-driven Models," AIAA Paper 2017–0314, 2017. https://doi.org/10.2514/6.2017-0314.

[29] Holland, J. R., Baeder, J. D., and Duraisamy, K., "Towards Integrated Field Inversion and Machine Learning With Embedded Neural Networks for RANS Modeling," AIAA Paper 2019-1884, 2019. https://doi.org/10.2514/6.2019-1884.

[30] Ho, J., and West, A., "Field Inversion and Machine Learning for turbulence modelling applied to three-dimensional separated flows," AIAA Paper 2021–2903, 2021. https://doi.org/https://doi.org/10.2514/6.2021-2903.

[31] Yan, C., Li, H., Zhang, Y., and Chen, H., "Data-driven turbulence modeling in separated flows considering physical mechanism analysis," , 2021.

[32] Jäckel, F., "A Closed-form Correction for the Spalart-Allmaras Turbulence Model for Separated Flow," AIAA Paper 2022–0462, 2022. https://doi.org/10.2514/6.2022-0462.

[33] Reed, W., and Hill, T., "Triangular Mesh Methods for the Neutron Transport Equation," Los Alamos Scientific Laboratory Technical Report LA-UR-73-479, 1973.

[34] Cockburn, B., and Shu, C.-W., "Runge-Kutta discontinuous Galerkin methods for convection-dominated problems," *Journal of Scientific Computing*, Vol. 16, No. 3, 2001, pp. 173–261. https://doi.org/https://doi.org/10.1023/A:1012873910884.

[35] Fidkowski, K. J., Oliver, T. A., Lu, J., and Darmofal, D. L., "*p*-Multigrid solution of high–order discontinuous Galerkin discretizations of the compressible Navier-Stokes equations," *Journal of Computational Physics*, Vol. 207, 2005, pp. 92–113. https://doi.org/10.1016/j.jcp.2005.01.005.

[36] Allmaras, S., Johnson, F., and Spalart, P., "Modifications and Clarifications for the Implementation of the Spalart-Allmaras Turbulence Model," Seventh International Conference on Computational Fluid Dynamics (ICCFD7) 1902, 2012.

[37] Ceze, M. A., and Fidkowski, K. J., "High-Order Output-Based Adaptive Simulations of Turbulent Flow in Two Dimensions," AIAA Paper 2015–1532, 2015. https://doi.org/10.2514/6.2015-1532.

[38] Fidkowski, K. J., "Three-Dimensional Benchmark RANS Computations Using Discontinuous Finite Elements on Solution-Adapted Meshes," AIAA Paper 2018–1104, 2018. https://doi.org/10.2514/6.2018-1104.

[39] Spalart, P. R., "Detached-Eddy Simulation," *Annual Review of Fluid Mechanics*, Vol. 41, 2009, pp. 181 – 202. https://doi.org/10.1146/annurev.fluid.010908.165130.

[40] Fidkowski, K. J., "Output error estimation strategies for discontinuous Galerkin discretizations of unsteady convection-dominated flows," *International Journal for Numerical Methods in Engineering*, Vol. 88, No. 12, 2011, pp. 1297–1322. https://doi.org/10.1002/nme.3224.

[41] Roe, P., "Approximate Riemann solvers, parameter vectors, and difference schemes," *Journal of Computational Physics*, Vol. 43, 1981, pp. 357–372. https://doi.org/https://doi.org/10.1016/0021-9991(81)90128-5.

[42] Bassi, F., and Rebay, S., "Numerical evaluation of two discontinuous Galerkin methods for the compressible Navier-Stokes equations," *International Journal for Numerical Methods in Fluids*, Vol. 40, 2002, pp. 197–207. https://doi.org/https://doi.org/10.1002/fld.338.

[43] Ceze, M. A., and Fidkowski, K. J., "Constrained pseudo-transient continuation," *International Journal for Numerical Methods in Engineering*, Vol. 102, 2015, pp. 1683–1703. https://doi.org/10.1002/nme.4858.

[44] Saad, Y., and Schultz, M. H., "GMRES: A Generalized Minimal Residual Algorithm for Solving Nonsymmetric Linear Systems," *SIAM Journal on Scientific Computing*, Vol. 7, No. 3, 1986, pp. 856–869. https://doi.org/https://doi.org/10.1137/0907058.

[45] Persson, P.-O., and Peraire, J., "Newton-GMRES Preconditioning for Discontinuous Galerkin Discretizations of the Navier-Stokes Equations," *SIAM Journal on Scientific Computing*, Vol. 30, No. 6, 2008, pp. 2709–2733. https://doi.org/https://doi.org/10.1137/070692108.

[46] Cash, J., "The integration of stiff initial value problems in ODEs using modified extended backward differentiation formulae," *Computers & mathematics with applications*, Vol. 9, No. 5, 1983, pp. 645–657. https://doi.org/https://doi.org/10.1016/0898-1221(83)90122-0.

[47] Fidkowski, K. J., and Darmofal, D. L., "Review of Output-Based Error Estimation and Mesh Adaptation in Computational Fluid Dynamics," *AIAA Journal*, Vol. 49, No. 4, 2011, pp. 673–694. https://doi.org/10.2514/1.J050073.

[48] J. E. Dennis, J., and More, J. J., "Quasi-Newton Methods, Motivation and Theory," *Society for Industrial and Applied Mathematics Review*, Vol. 19, 1977, pp. 359 – 372. https://doi.org/10.1137/1019005.

[49] Rothacker, B. A., Ceze, M. A., and Fidkowski, K. J., "Adjoint-based error estimation and mesh adaptation for problems with output constraints," AIAA Paper 2014–2576, 2014. https://doi.org/10.2514/6.2014-2576.

[50] Persson, P.-O., and Peraire., J., "Sub-cell shock capturing for discontinuous Galerkin methods," AIAA Paper 2006-112, 2006. https://doi.org/https://doi.org/10.2514/6.2006-112.

[51] Shimizu, Y. S., and Fidkowski, K. J., "Output Error Estimation for Chaotic Flows," AIAA Paper 2016-3806, 2016. https://doi.org/https://doi.org/10.2514/6.2016-3806.

[52] Spalart, P., and Shur, M. L., "On the Sensitization of Turbulence Models to Rotation and Curvature," *Aerospace Science and Technology*, Vol. 1, No. 5, 1997, pp. 297–302. https://doi.org/10.1016/S1270-9638(97)90051-1.

[53] Werbos, P., "Beyond Regression: New Tools for Prediction and Analysis in the Behavioral Science," Ph.D. thesis, Harvard University, 1974.

[54] Barron, A. R., "Approximation and estimation bounds for artificial neural networks," *Machine Learning*, Vol. 14, No. 1, 1994, pp. 115–133. https://doi.org/https://doi.org/10.1007/BF00993164.

[55] Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mané, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viégas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., and Zheng, X., "TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems," , 2015. URL http://tensorflow.org/, software available from tensorflow.org.

[56] Krakos, J. A., and Darmofal, D. L., "Effect of Small-Scale Unsteadiness on Adjoint-Based Output Sensitivity," AIAA Paper 2009-4274, 2009. https://doi.org/https://doi.org/10.2514/6.2009-4274.

[57] Krakos, J. A., Wang, Q., Hall, S. R., and Darmofal, D. L., "Sensitivity analysis of limit cycle oscillations," *Journal of Computational Physics*, Vol. 231, No. 8, 2012, pp. 3228–3245. https://doi.org/https://doi.org/10.1016/j.jcp.2012.01.001.