33RD CONGRESS
OF THE INTERNATIONAL COUNCIL
OF THE AERONAUTICAL SCIENCES
STOCKHOLM, SWEDEN, 4–9 SEPTEMBER, 2022

ICAS 2022 SWEDEN

DATA-DRIVEN ADAPTATION AND OPTIMIZATION OF TURBULENT FLOWS

Krzysztof J. Fidkowski[1]

[1]University of Michigan, Ann Arbor, MI, USA

Abstract

We present a data-driven approach for calculating adjoint sensitivities in unsteady turbulent flows, with application to shape optimization and output-based adaptation. The approach does not use unsteady adjoint equations, which are expensive to solve and become unstable for chaotic problems, but instead relies on unsteady data to train a corrected turbulence model, which then yields the required adjoint solutions. It is non-intrusive and inexpensive, requiring only a small number of unsteady forward simulations, but sufficiently powerful to capture unsteady effects in the sensitivities. Results for high-order discretizations of the unsteady Navier-Stokes equations, augmented by a corrected Spalart-Allmaras turbulence closure, demonstrate the ability of the approach in driving airfoil shape optimization and in adapting unsteady flowfields to target statistical outputs of interest.

Keywords: Optimization, Adaptation, High-Order, Field Inversion, Machine Learning

1 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 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. Ensuring the accuracy of such simulations and enabling their use in design are the topics addressed in this work.

A persistent theme of this paper is the calculation of sensitivities in unsteady flowfields using adjoints. Adjoint methods, which enjoy wide popularity in shape optimization [41, 35, 33, 12, 10] and output-based mesh adaptation [5, 48, 36, 18], cannot be directly applied to unsteady turbulent flow simulations, as they are unstable for such systems [26] and require expensive regularization techniques to provide meaningful answers [49, 51, 50, 7, 37, 38]. Furthermore, unsteady adjoints come with massive storage and computation requirements, which become impractical for forward simulations that already stress computational resources [21]. 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 [34, 32, 19, 14, 27, 16, 39, 6, 2]. However, for turbulent flows, they provide potentially 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 [29]. Instead, of engineering interest are generally only statistical quantities, such as time averages of outputs, which should be predictable by steady-state models.

Many steady-state turbulence models based on Reynolds averaging of the Navier-Stokes equations exist but often do not provide sufficient accuracy relative to unsteady models at off-design conditions. Instead of choosing one, we create multiple steady-state models, each specific to the unsteady case being considered. This approach is based on the idea of field inversion and machine learning (FIML) [40, 47, 23, 22, 52, 25], which starts with an existing steady-state model and corrects it via a PDE-level multiplicative factor on the turbulence production term. A high-fidelity unsteady simulation 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.

The proposed approach consists of iterative turbulence model calibration, through FIML and a small number of unsteady simulations, coupled with steady-state optimization and adaptation using the corrected turbulence models. The number of unsteady simulations required is much lower than the function evaluations demanded by an iterative optimization or adaptation approach, as these iterations are 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 are typically required. However, as will be shown in the results, the accuracy improvements of the corrected model make a marked difference in the optimized shapes and meshes, particularly in cases where the uncorrected model looses applicability.

The outline for the remainder of this paper is as follows. Section 2 presents the numerical approach used in this work, a discontinuous finite-element discretization for both the primal and adjoint problems. Section 3 presents the design parametrization for airfoil shapes, and the gradient-based optimization method. Section 4 presents the field-inversion and machine-learning algorithm, with a novel objective function geared for output error estimation. Sections 5 and 6 discuss the error estimation, based on the adjoint-weighted residual, and the adaptation methods, which include both h and p refinement. Finally, Section 7 presents optimization and adaptation results, and Section 8 concludes with a summary and a discussion of future directions.

2 Discretization

2.1 Governing Equations

The Reynolds-averaged Navier-Stokes (RANS) equations can be written as

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

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 from from the turbulence model, in this work the Spalart-Allmaras (SA) closure [3, 9]. A detailed exposition of the equations can be found in previous work [9, 17]. The eddy viscosity equation contains the turbulence production term that in this work is modified through a multiplicative correction factor, β,

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

In this equation, ρ is the density, $\vec{\mathbf{v}}$ is the velocity, $\tilde{\nu}$ turbulence working variable, f_n, c_{b2}, σ are model functions/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.

2.2 The Discontinuous Galerkin Method

We use a discontinuous Galerkin (DG) discretization [42, 11, 20, 13], with the Roe [43] convective flux and the second form of Bassi and Rebay (BR2) [4] for the viscous treatment. The state is

approximated on an unstructured mesh of non-overlapping elements using polynomials of order p. The semi-discretized form of the equations is

$$\mathbf{M}\frac{d\mathbf{U}}{dt}+\mathbf{R}(\mathbf{U})=\mathbf{0}, \tag{3}$$

where $\mathbf{U}\in\mathbb{R}^N$ is the discrete state vector, N is the total number of unknowns, $\mathbf{R}(\cdot)\in\mathbb{R}^N$ 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. The implicit solver consists of a Newton-Raphson method with the generalized minimum residual (GMRES) [45] linear solver, preconditioned by an iterative smoother, and for unsteady simulations, we use a third-order modified extended backward difference formula [8] applied to the semi-discrete form.

2.3 The Discrete Adjoint

For a scalar output computed from the state vector, $J(\mathbf{U})$, the discrete steady adjoint vector, $\Psi\in\mathbb{R}^N$, is the local sensitivity of J to perturbations in the steady residual, $\mathbf{R}$ [18]. It satisfies the adjoint equation,

$$\left(\frac{\partial\mathbf{R}}{\partial\mathbf{U}}\right)^T\Psi+\left(\frac{\partial J}{\partial\mathbf{U}}\right)^T=\mathbf{0}, \tag{4}$$

which is solved using the same 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,

$$\frac{dJ}{d\mathbf{x}}=\frac{\partial J}{\partial\mathbf{x}}+\Psi^T\frac{\partial\mathbf{R}}{\partial\mathbf{x}}. \tag{5}$$

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.

3 Design Optimization

We consider trimmed optimization problems, in which shape design parameters, $\mathbf{x}^{\text{des}}\in\mathbb{R}^{n_{\text{des}}}$, and trim parameters, $\mathbf{x}^{\text{trim}}\in\mathbb{R}^{n_{\text{trim}}}$, are 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

$$\begin{aligned}\min_{\mathbf{x}^{\text{des}}}\quad & J(\mathbf{U},\mathbf{x})\\ \text{s.t.}\quad & \mathbf{R}(\mathbf{U},\mathbf{x})=\mathbf{0}\\ & \mathbf{R}^{\text{trim}}\equiv\mathbf{J}^{\text{trim}}(\mathbf{U},\mathbf{x})-\bar{\mathbf{J}}^{\text{trim}}=\mathbf{0}\end{aligned} \tag{6}$$

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.

To solve Eqn. 6, we use the Broyden-Fletcher-Goldfarb-Shanno (BFGS) [24] 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 [44]

$$\mathbf{x}^{\text{trim}}_{k+1}=\mathbf{x}^{\text{trim}}_k+\Delta\mathbf{x}^{\text{trim}}_k,\qquad \Delta\mathbf{x}^{\text{trim}}_k=\left[\frac{d\mathbf{J}^{\text{trim}}}{d\mathbf{x}^{\text{trim}}}\right]^{-1}_k\left(\bar{\mathbf{J}}^{\text{trim}}-\mathbf{J}^{\text{trim}}(\mathbf{U}_k,\mathbf{x}_k)\right), \tag{7}$$

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, where at each iteration, the trimmed gradient of the objective with respect to the design parameters is

$$\left.\frac{dJ}{d\mathbf{x}^{\text{des}}}\right|_{\text{trim}}=\frac{dJ}{d\mathbf{x}^{\text{des}}}+\frac{dJ}{d\mathbf{x}^{\text{trim}}}\frac{d\mathbf{x}^{\text{trim}}}{d\mathbf{x}^{\text{des}}},\qquad \frac{d\mathbf{x}^{\text{trim}}}{d\mathbf{x}^{\text{des}}}=-\left[\frac{d\mathbf{J}^{\text{trim}}}{d\mathbf{x}^{\text{trim}}}\right]^{-1}\frac{d\mathbf{J}^{\text{trim}}}{d\mathbf{x}^{\text{des}}}, \tag{8}$$

The gradient of the objective output as given in Eqn. 8 is then used in the BFGS algorithm. During the BFGS line search, each objective function evaluation accounts for a linearized trim parameter change and includes an off-trim correction, calculated from the trim and output adjoints.

We optimize airfoils with camber and thickness profiles parametrized via polynomial expressions with bounding multiplicative envelopes,

$$z_c(x) = x(1-x)\left[c_0 + c_1 x + \ldots + c_{p_c} x^{p_c}\right], \tag{9}$$

$$t(x) = a\ \mathrm{abs}_{\mathrm{smooth}}\left\{\sqrt{x}(1-x)\left[1 + t_1 x + \ldots + t_{p_t} x^{p_t}\right],\ 0.2x(1-x)\right\} \tag{10}$$

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

$$\mathrm{abs}_{\mathrm{smooth}}(a,b) = \begin{cases} |a| & \text{if } |a| \geq b \\ (a^2+b^2)/(2b) & \text{otherwise} \end{cases} \tag{11}$$

The coefficient a enforces a constant airfoil area, A,

$$\int_0^1 t(x)\,dx = A. \tag{12}$$

4 Field Inversion and Machine Learning

From an unsteady simulation, we compute a time-average state

$$\bar{\mathbf{U}} = \frac{1}{T_f - T_i}\int_{T_i}^{T_f} \mathbf{U}(t)\,dt, \tag{13}$$

with initial time T_i chosen after transients, and final time T_f sufficiently long to ensure adequate statistics. The goal in field inversion and machine learning (FIML) is to solve an inverse problem for a correction field that makes the corrected turbulence model solution close to $\bar{\mathbf{U}}$, and to train a machine-learning model to calculate the correction field as a function of local state information.

4.1 Field Inversion

The scalar correction field, $\beta(\vec{x})$, with a nominal value of 1, multiplies the production term in the eddy-viscosity transport equation of the SA turbulence model. Approximating the field using a Lagrange DG basis, here $p = 1$, field inversion can be written as a constrained discrete minimization problem,

$$\begin{aligned} \min_{\beta} \quad & J^{\mathrm{inv}} \equiv \mathscr{F}(\mathbf{U}(\beta)) + \frac{\gamma}{2}(\beta-1)^T\mathbf{M}(\beta-1) \\ \text{s.t.} \quad & \mathbf{R}(\mathbf{U},\beta) = \mathbf{0} \end{aligned} \tag{14}$$

where $\mathscr{F}(\mathbf{U},\bar{\mathbf{U}})$ measures the error in the solution relative to the unsteady average and is supplemented by a regularization term, here with $\gamma = 10^{-7}$.

The baseline field inversion error measure is an integral of the surface stress distribution error,

$$\mathscr{F}^{\mathrm{dist}}(\mathbf{U},\bar{\mathbf{U}}) = \frac{1}{2}\int_{\mathrm{airfoil}} \|\boldsymbol{\sigma}(\mathbf{U})\cdot\vec{n} - \boldsymbol{\sigma}(\bar{\mathbf{U}})\cdot\vec{n}\|^2\,ds, \tag{15}$$

where σ is the stress tensor and $\vec{n}$ is the unit normal vector on the airfoil surface. In output-based error estimation, we also use a measure that mimics the adjoint-weighted residual,

$$\mathscr{F}^{\mathrm{AWR}}(\mathbf{U},\bar{\mathbf{U}}) = \sum_e \frac{1}{2}\left(\mathbf{W}_e^T(\mathbf{U}_e(\beta) - \bar{\mathbf{U}}_e)\right)^2 \qquad \mathbf{W}_e^T \equiv \boldsymbol{\Psi}_e^T\frac{\partial \mathbf{R}_e}{\partial \mathbf{U}_e}. \tag{16}$$

Using either of the two error measures, 15 or 16, the inverse problem is solved using a limited-memory Broyden–Fletcher–Goldfarb–Shanno (L-BFGS) algorithm [30], with a history of 10 updates for approximating the inverse Hessian. The required gradient of J^{inv} with respect to β is evaluated using an adjoint, while the residual linearization with respect to β is calculated using a small number of inexpensive finite differences.

4.2 Machine Learning

While field inversion produces a correction field for a particular mesh and shape, adaptation and optimization require corrections over different meshes and geometries. The second step therefore consists of training a local model to produce the correction from the average solution. In FIML, this model is an artificial neural network that maps local states and geometry information to the correction field [40]. Figure 1 shows the structure of the network used in this work, a single-hidden-layer perceptron that maps the feature vector, $\mathbf{x}_0 \in \mathbb{R}^{n_0}$, to the correction scalar, β, through the hidden layer, $\mathbf{x}_1 \in \mathbb{R}^{n_1}$. The parameters associated with the network consist of the weights and biases,

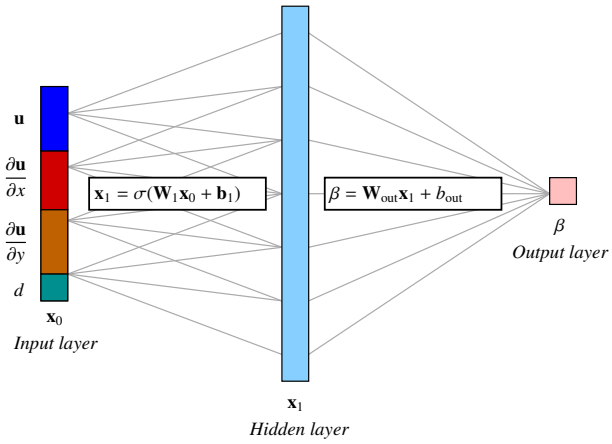


Figure 1 – Structure of the artificial neural networks used to predict the correction field.

$\mathbf{W}_i \in \mathbb{R}^{n_i \times n_{i-1}}$, $\mathbf{b}_i \in \mathbb{R}^{n_i}$, where n_i is the number of neurons in layer i. An entry-wise sigmoid activation function, $\sigma(x) = 1/(1+e^{-x})$, is applied to the map from the input layer to the hidden layer. The size of the input layer is $n_0 = (1+\dim)s+1$ neurons (state, its gradient, wall distance), and the size of the hidden layer is set to $n_1 = 30$ for the two-dimensional problems considered in this work.

The network parameters are optimized using an adaptive moment (Adam) estimation algorithm in TensorFlow [1], with a mean squared error loss between the predicted and actual output-layer values, measured at quadrature points of each element. The network is then implemented as a physics model in the turbulence source calculation, which affects the residual and hence the primal and adjoint solutions. The network is linearized in the calculation of the residual Jacobian matrix for the Newton and discrete-adjoint solvers.

5 Output-Based Error Estimation

For a time-averaged output computed from the unsteady discrete state,

$$\bar{J} \equiv \frac{1}{T_f - T_i} \int_{T_i}^{T_f} J(\mathbf{U}(t))\, dt, \tag{17}$$

the unsteady adjoint, $\Psi(t)$, is the sensitivity of $\bar{J}$ to residual perturbations, which here are assumed to be predominantly spatial. Decomposing the adjoint and residual perturbation into time-averaged and time-varying components, and assuming that the adjoint and residual are not strongly correlated in time, we obtain an adjoint-weighted residual error estimate [13, 31, 16],

$$\delta \bar{J} \approx \bar{\Psi}^{\mathsf{T}} \delta \bar{\mathbf{R}}, \tag{18}$$

where $\bar{\cdot}$ denotes time averaging. The solution of the augmented (corrected) steady-state system yields an approximation of the time-averaged adjoint, whereas the residual perturbation is computed by time-averaging the spatial residual error between a coarse (H) and a fine (h) space. The output error estimate therefore reads

$$\delta \bar{J}_h \approx \delta \bar{\Psi}_h^{\mathsf{T}} \delta \bar{\mathbf{R}}_h, \quad \delta \bar{\mathbf{R}}_h \equiv \frac{1}{T_f - T_i} \int_{T_i}^{T_f} \delta \mathbf{R}_h(t)\, dt, \quad \delta \mathbf{R}_h(t) = -\mathbf{R}_h(\mathbf{U}_h^H(t)) + \mathbf{M}_h \mathbf{I}_h^H \mathbf{M}_H^{-1} \mathbf{R}_H(\mathbf{U}_H(t)), \tag{19}$$

where $\mathbf{U}_h^H(t) = \mathbf{I}_h^H \mathbf{U}_H(t)$ the prolongation of the discrete coarse unsteady state $\mathbf{U}_H(t)$ into the fine space. The error estimate uses adjoint perturbation instead of the fine-space adjoint itself in order to reduce errors arising from lack of strict Galerkin orthogonality in the residuals [27].

The error estimate requires the solution of the augmented system, here obtained from the FIML approach. The field inversion and machine-learning rely on an unsteady simulation for the training data, which is the expensive part of the process, but it is only a primal solution. An unsteady adjoint is not required for the error estimate.

6 Mesh Adaptation

The error estimate in Eqn. 19 is calculated after an unsteady simulation, which provides the time-averaged residual and FIML training data. In order to adapt, the error estimate is localized to elements,

$$\mathscr{E}_e \equiv \left| \delta\Psi_{he}^{\mathsf{T}} \delta\mathbf{R}_{he} \right|, \qquad \delta J^{\text{cons}} \equiv \sum_e \mathscr{E}_e, \tag{20}$$

where the subscript e denotes degrees of freedom associated with element e. The above equation also defines a *conservative* error estimate, δJ^{cons}, as the sum of the error indicators.

In this work, three adaptive strategies are considered: output-based p-adaptation, residual-based p-adaptation, and output-based mesh optimization. The first two rely solely on an elemental error indicator and follow a fixed-fraction strategy. The third is mesh optimization through error sampling and synthesis, MOESS [54, 53, 15], which optimizes the mesh using anisotropic metric-based adaptation, by minimizing the output error at a prescribed computational cost. With the latter, even by taking just one MOESS adaptation iteration per unsteady run, rapid convergence to the optimal mesh has been observed, so that only 3-4 unsteady simulations are typically required.

Figure 2 illustrates the unsteady adaptation loop, which starts with a given mesh/order distribution from a steady adaptation/optimization using the augmented but uncorrected model. An unsteady simulation on the current mesh provides the output and target data for field inversion. Machine learning converts the correction field, $\beta(\vec{x})$, into a neural network model for β in terms of the state, its gradient, and the wall distance, which is then used in the corrected-model simulation. The adjoint from this simulation is used together with the time-averaged residual to calculate the output-error estimate and adaptive indicator, which then drive mesh adaptation. The process repeats with an unsteady simulation on the adapted mesh.

7 Results

This section presents an optimization result and an adaptation result for two unsteady aerodynamics problems. Discrete solutions are obtained using $p = 2$ approximation, except for the p-refinement study. Both problems are two-dimensional but share features of more complex, and expensive, three-dimensional turbulent simulations.

7.1 Mesh Adaptation: NACA 0012 Airfoil at $\alpha = 7^\circ$, $Re = 10{,}000$

This adaptive result focuses on efficacy of the error estimate and the convergence of the time-averaged outputs with respect to cost, as measured by spatial degrees of freedom. The temporal discretization remains fixed at a high resolution, determined empirically such that temporal errors do not pollute the results relative to spatial errors. For clarity, the adopted naming convention for the adaptive strategies is as follows: FIML MOESS [AWR] is mesh optimization at a given target `dof` using the FIML-based approach for the adjoint and field-inversion objective $\mathscr{F}^{\text{dist}}$ (default) or $\mathscr{F}^{\text{AWR}}$ (AWR); RANS MOESS is mesh optimization at a given target `dof` using the baseline, uncorrected RANS model; FIML p-adapt [output] is p refinement at a fixed fraction f^{adapt} using the FIML adjoint and error estimate E1; p-adapt residual is p refinement using the unweighted unsteady residual.

The example considers flow governed by the compressible Navier-Stokes equations over a NACA 0012 airfoil, at Mach number $M = 0.2$, angle of attack $\alpha = 7^\circ$, and a relatively low Reynolds number

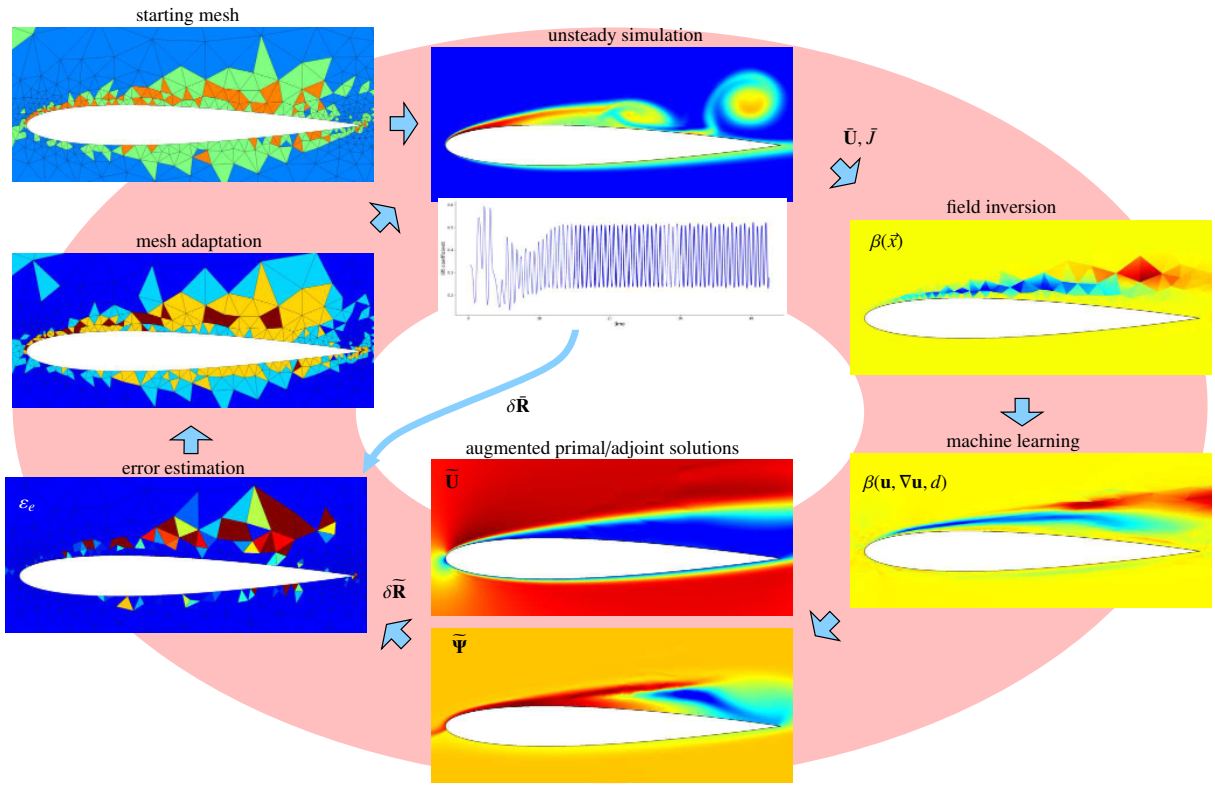
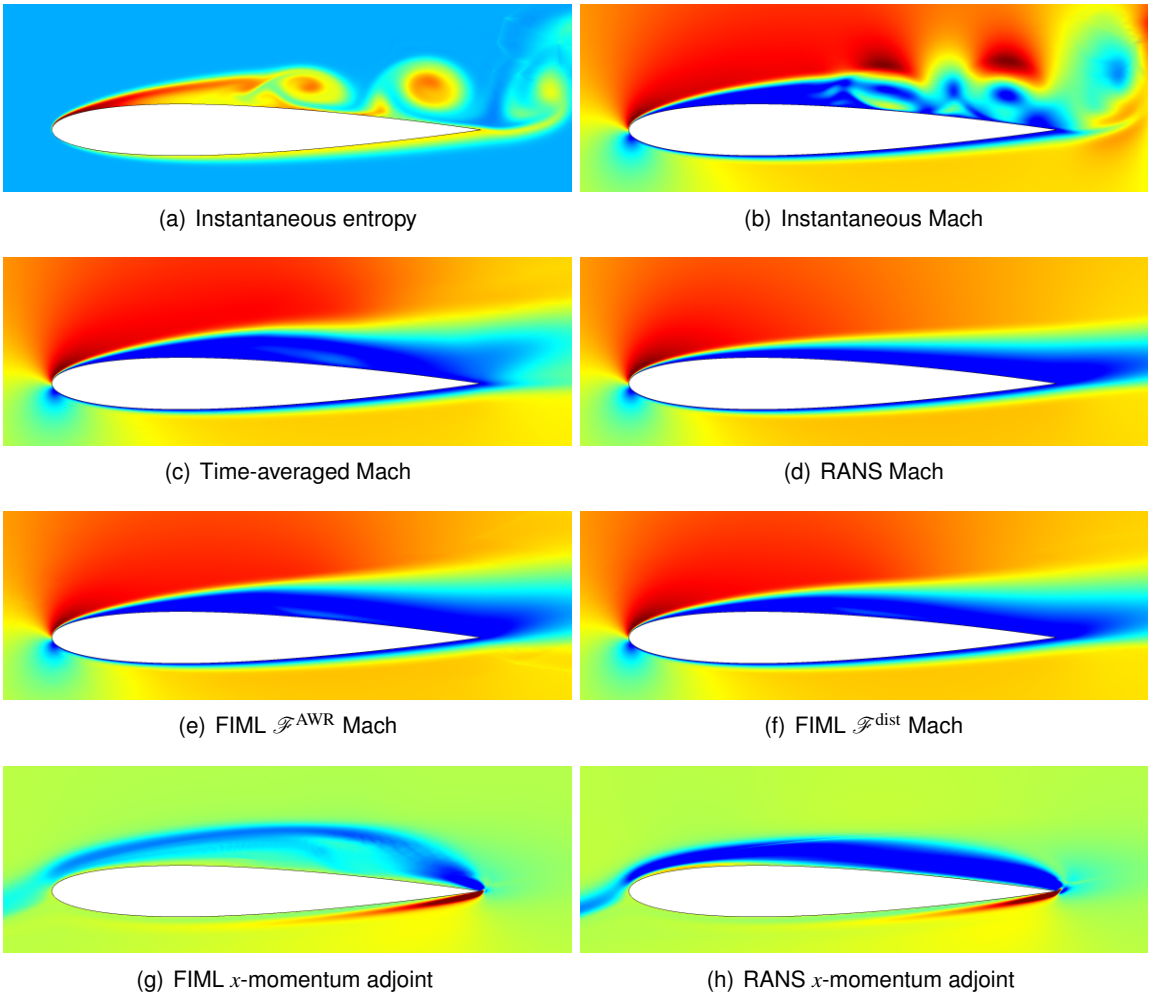


Figure 2 – Flowchart of the unsteady mesh-adaptation process.

$Re = 10^4$. The computational domain is meshed with unstructured triangles that are curved at the airfoil surface using a cubic mapping. The flow consists of a thin laminar boundary layer on the lower surface and a thicker layer on the upper surface that breaks down into unsteady vortices. Figure 3 shows instantaneous snapshosts of the solution. This figure also shows the time-averaged Mach contours, which exhibit a large separation bubble structure on the upper surface. The uncorrected RANS solution, also shown in the figure, fails to predict this structure and instead exhibits a thinner, mostly attached boundary layer. In contrast, the corrected RANS fields, obtained using FIML, more closely represent the time-averaged state. A difference also exists between the field inversion error measures: $\mathscr{F}^{\text{AWR}}$ yields a more accurate fowfield compared to $\mathscr{F}^{\text{dist}}$, due to the former's accounting of errors arising from the domain interior. Figure 3 also compares the lift adjoint fields for RANS versus FIML: these differ because they are based on a different baseline flow state, and because the residual Jacobian matrix includes a linearization of the machine-learning model.

Figure 4 shows the results of the adaptations, plotted as the output of interest, the average lift coefficient in this case, versus spatial degrees of freedom, which is the cost measure. The error estimates are obtained using the conservative sum of indicators, Eqn. 20. These are shown as shaded bands around the outputs, at $\pm\delta J^{\text{cons}}$. The exact solution is obtained from a $p = 3$ simulation on a uniformly-refined version of the finest lift-adapted mesh.

From Figure 4, we see that uncorrected RANS converges quickly but severely under-predicts the lift output. On the other hand, the unsteady output-adaptive approaches perform much better. In particular, mesh optimization yields very good results, even with coarse meshes: e.g. at $3000\texttt{dof}$, the meshes contain only 500 elements. The only other adaptive approach that comes close is lift-based p refinement, which eventually jumps close to the correct average value by the third adaptive iteration. However, local p refinement cannot be done haphazardly, as demonstrated by the poor performance of the unweighted residual-based indicator, which does not get close to the correct output due to its preoccupation with large residuals in areas that minimally impact the lift output. In addition, p refinement is sensitive to the initial mesh, which in this case already has reasonable refinement in areas such as the leading and trailing edge.



(a) Instantaneous entropy

(b) Instantaneous Mach

(c) Time-averaged Mach

(d) RANS Mach

(e) FIML $\mathscr{F}^{\mathrm{AWR}}$ Mach

(f) FIML $\mathscr{F}^{\mathrm{dist}}$ Mach

(g) FIML x-momentum adjoint

(h) RANS x-momentum adjoint

Figure 3 – Instantaneous and time-averaged entropy, Mach number, and adjoint contours for flow over a NACA 0012 airfoil at $M = 0.2, Re = 1 \times 10^4, \alpha = 7^\circ$.

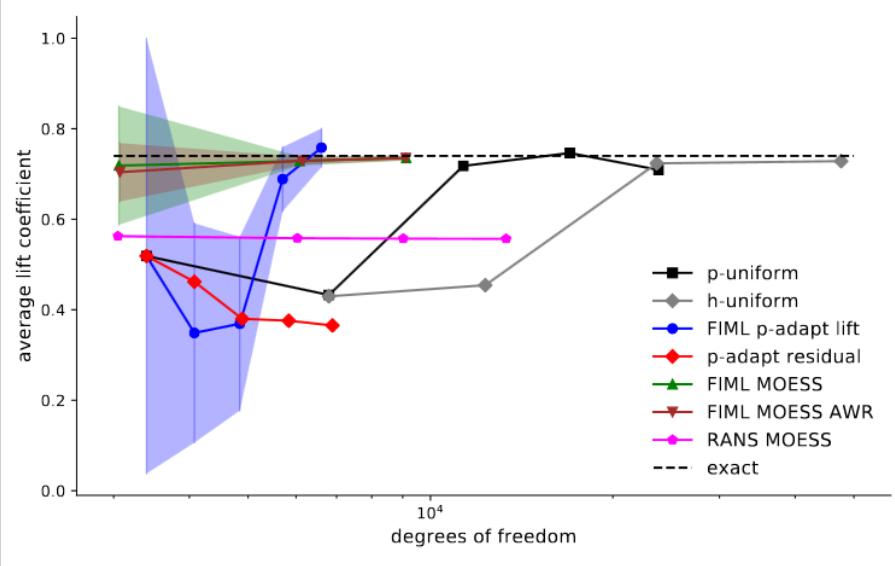


Figure 4 – Lift coefficient convergence histories for various adaptive methods applied to the NACA 0012 airfoil at $M = 0.2, Re = 10^4, \alpha = 7^\circ$.

Figure 5 shows the output history for lift-based p-adaptation applied to this case. The multiple unsteady runs are shown sequentially, so that the horizontal axis indicates the total unsteady run time, as measured in simulation units, where one time unit is the airfoil chord divided by the freestream speed. Each unsteady run includes a "burn" time followed by a time-averaging window. At the initial $p = 1$ uniform order, the approximation space is too coarse to resolve any unsteady behavior, and hence the initial flat output history. Note that the initial mesh only has 1133 elements. The green bars at the end of each unsteady run represent the $\pm\delta J^{\text{cons}}$ error estimates, obtained using the adjoint-weighted residual. The estimates are reasonably accurate, and adaptation improves the average output with only localized high-order refinement. In comparison to lift-based p-adaptation, residual-based p-adaptation does not perform very well, as residuals remain large away from the airfoil, particularly on the larger elements.

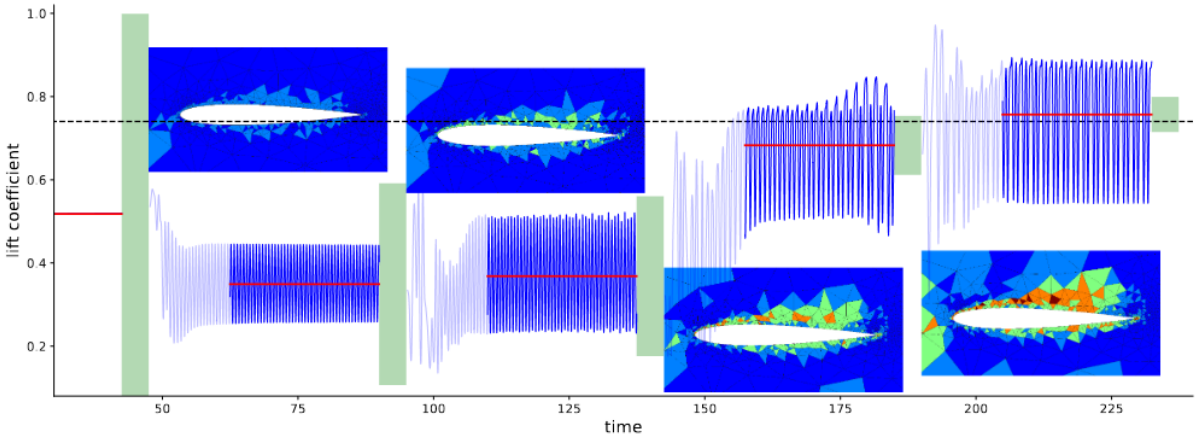


Figure 5 – p-adaptation history for the NACA 0012 airfoil at $M = 0.2, Re = 10^4, \alpha = 7^\circ$. The order field range is $p = 1$ (blue) to $p = 5$ (red). Red lines denote time-average outputs, and the green bars are error estimates at $\pm\delta J^{\text{cons}}$.

Turning to FIML MOESS, Figure 6 shows the output history for lift adaptation at $6000\texttt{dof}$. The starting mesh is the RANS mesh, also at $6000\texttt{dof}$. The unsteady output is severely under-predicted by an unsteady simulation on the initial RANS mesh, even though the resolution does not appear egregiously incorrect in any area. In just one unsteady adaptation iteration, however, a redistribution of the mesh resolution dramatically improves the unsteady output, and additional iterations bring about smaller changes. The error estimate under-predicts the actual error on the coarse mesh, which may be of insufficient resolution to obtain accurate errors. On subsequent iterations, the error estimates become accurate.

Table 1 presents the outputs, error estimates, and actual errors for various FIML MOESS adaptive simulations. The outputs and error estimates were obtained by averaging the results of the latter half of the MOESS iterations. We see a general trend of decreasing actual errors and improved error estimate efficacy with increasing degrees of freedom. Comparing the two FIML inversion error measures, using $\mathscr{F}^{\text{AWR}}$ leads to tighter error estimates, although the actual errors do not change much from the baseline $\mathscr{F}^{\text{dist}}$ case. The tighter error estimates could be due to improved FIML domain-interior solutions with $\mathscr{F}^{\text{AWR}}$ producing a more accurate adjoint field.

Finally, Figure 7 compares four adapted meshes, each at the final MOESS iteration. The distribution of mesh resolution and anisotropy reflects some of the solution features shown in Figure 3. First, the RANS-adapted mesh contains elements of relatively high anisotropy along the upper surface, mostly close to the surface and at the edge of the boundary layer. In contrast, meshes obtained using MOESS driven by the presented unsteady error estimates exhibit thicker boundary-layer adaptation, with more isotropic elements and localized refinement approximately half-way down the chord on the upper surface. In this region, the laminar flow off the leading edge begins to break down into unsteady vortices. The output-adaptive approaches tag this region as important for an accurate average force

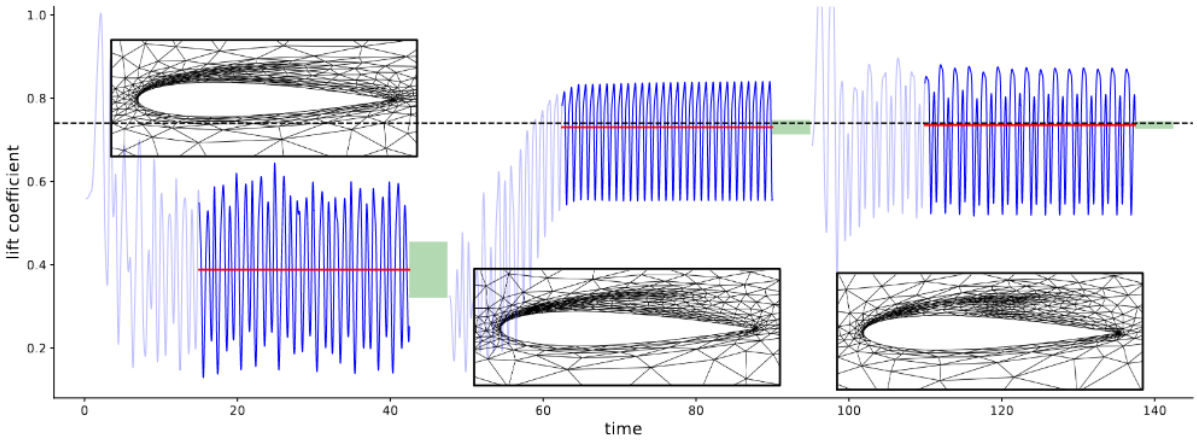
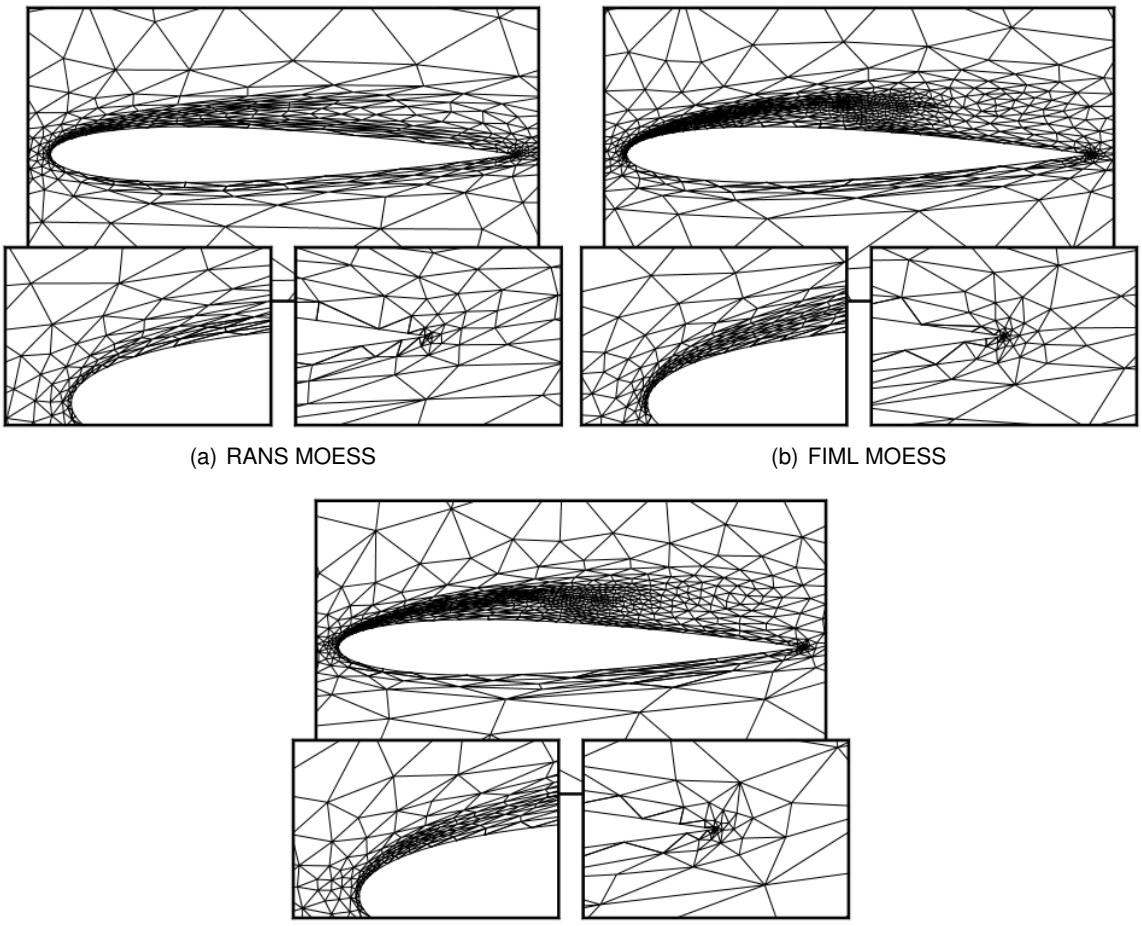


Figure 6 – FIML MOESS adaptation history using $6000\texttt{dof}$ for the NACA 0012 airfoil at $M = 0.2, Re = 10^4, \alpha = 7^\circ$. Red lines denote time-averaged outputs, and the green bars are error estimates at $\pm\delta J^{\text{cons}}$.

Table 1 – Averaged lift coefficient and error estimates for FIML MOESS adaptive runs for the NACA 0012 airfoil at $M = 0.2, Re = 10^4, \alpha = 7^\circ$.

Target `dof`	FIML objective	Output	Error Estimate	Actual Error
3000	$\mathscr{F}^{\text{dist}}$	0.7186	0.1309	0.0214
3000	$\mathscr{F}^{\text{AWR}}$	0.7040	0.0645	0.0360
6000	$\mathscr{F}^{\text{dist}}$	0.7288	0.0099	0.0112
6000	$\mathscr{F}^{\text{AWR}}$	0.7294	0.0087	0.0106
9000	$\mathscr{F}^{\text{dist}}$	0.7355	0.0057	0.0045
9000	$\mathscr{F}^{\text{AWR}}$	0.7353	0.0051	0.0047

prediction on the airfoil. In addition, the output-adapted meshes contain more refinement close to the upper aft surface of the airfoil, likely due to the importance of capturing the interaction of the vortices and the boundary layer. The output adaptation also heavily targets the edge of the laminar region coming off the front of the airfoil, as the location of this feature dictates the outer flow over the airfoil, which strongly affects the lift.



(a) RANS MOESS (b) FIML MOESS

(c) FIML MOESS AWR

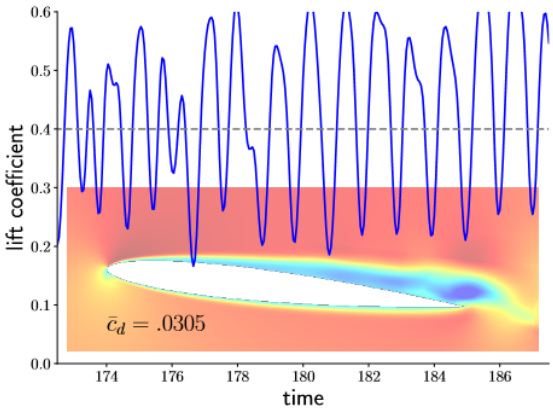
Figure 7 – Various adapted meshes at $9000\texttt{dof}$ for the NACA 0012 airfoil at $M = 0.2, Re = 10^4, \alpha = 7^\circ$.

7.2 Shape Optimization: Drag minimization at constant lift, $Re = 10{,}000$

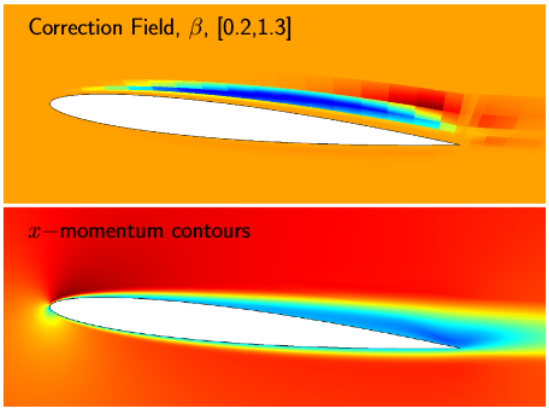
We now consider a drag minimization problem at a prescribed lift coefficient of $c_\ell = 0.4$, and $M = 0.2$, $Re = 10^4$. No turbulence model is used in the unsteady simulations at this low Reynolds number. The area of the airfoil is constrained to $A = .06c^2$ via Eqn. 12. Three camber parameters, c_0, c_1, c_2 in Eqn. 9, and two thickness parameters, t_1, t_2 in Eqn. 10, dictate the shape of the airfoil. All five of these parameters are set to zero to obtain the initial shape.

We first trim the initial airfoil in unsteady mode to determine the lift coefficient that attains the prescribed average lift coefficient. Figure 8(a) shows the lift coefficient time history for a portion of the trimmed condition, along 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.

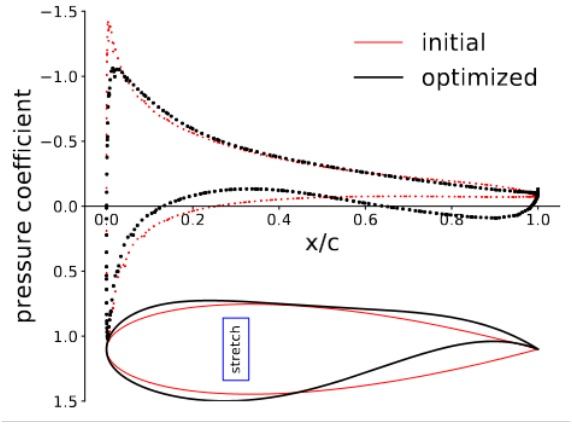
The oscillatory flowfield and outputs prevent the application of standard unsteady adjoint techniques for computing gradients [28, 46]. In addition to this instability, a full unsteady adjoint solution is also



(a) Unsteady solution



(b) Field inversion



(c) FIML Optimization

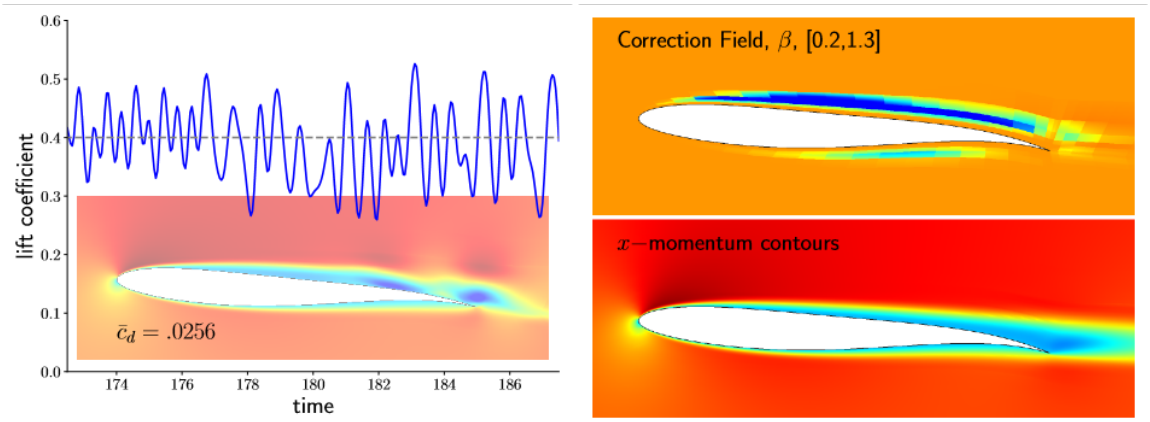
Figure 8 – First FIML optimization iteration for minimizing drag on an airfoil at $M = 0.2, Re = 10^4, c_\ell = 0.4$.

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 4. Figure 8(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.

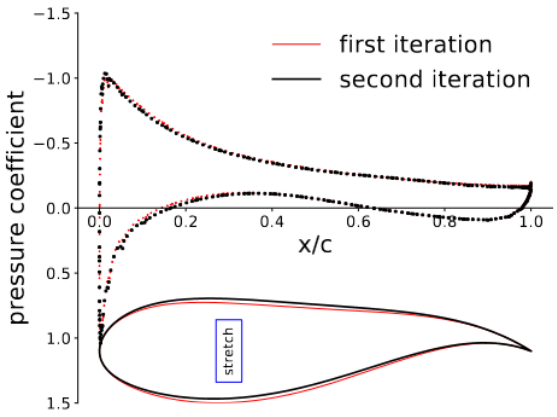
Data from this correction field were used to train a neural network model, which is then used in a steady shape optimization. Figure 8(c) shows the result of 50 BFGS optimization iterations using this first FIML model. The optimized shape exhibits a reduced suction peak, a higher aft-loading redistribution 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 9 shows the results of the second FIML optimization iteration, which parallel those of Figure 8.



(a) Unsteady solution

(b) Field inversion



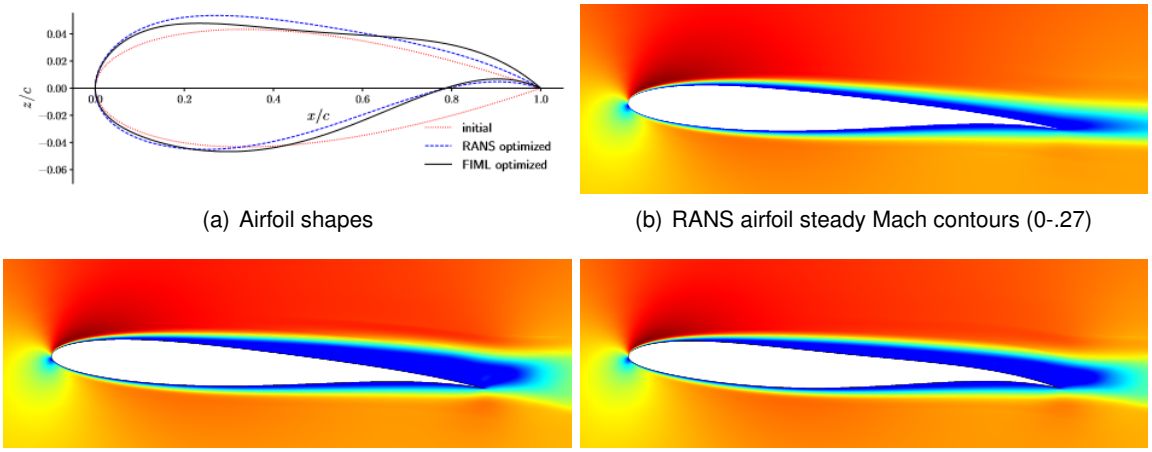
(c) FIML Optimization

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

The flowfield remains unsteady for the optimized airfoil, but the unsteady trimming time history in Figure 9(a) shows a reduced amplitude of the lift coefficient variation, by almost a factor of two

compared to Figure 8(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 9(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 9(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.



(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)

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

Two additional FIML optimizations were performed, each with a separate unsteady simulation, inversion, and steady optimization. Figure 10 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 10(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 10(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 2 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 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 10. 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.

Table 2 – 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, 1^{st} iteration	0.0256
FIML-optimization, 2^{nd} iteration	0.0241
FIML-optimization, 3^{rd} iteration	0.0215
FIML-optimization, 4^{th} iteration	0.0212

8 Conclusions

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

The adjoint calculation proposed in this work is instead based on a corrected RANS model obtained from unsteady simulation data. Following field inversion for the correction, 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. The network must be linearized for accurate adjoint calculations and is used only for a single iteration of adaptation or optimization.

The results show the ability of the adjoint from the corrected model to accurately predict residual sensitivities in the unsteady flows, leading to both optimal adapted meshes and shapes. These contrast with RANS-alone meshes and shapes, which yield sub-optimal performance. As the method is not intrusive into the unsteady simulation, independent "black-box" unsteady solvers could be used, with various levels of fidelity, including detached or large eddy simulations. Only average forward flowfield statistics are required for the inversion and training. Future work will investigate pushing the envelope of the FIML/RANS combination to stalled conditions and bluff-bodies.

9 Contact Author Email Address

Krzysztof Fidkowski: `kfid@umich.edu`

10 Copyright Statement

References

[1] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

[2] Frédéric Alauzet, Adrien Loseille, and Geraldine Olivier. Time accurate anisotropic goal-oriented mesh adaptation for unsteady flows. *Journal of Computational Physics*, 373(15):28–63, 2018.

[3] S.R. Allmaras, F.T. Johnson, and P.R. Spalart. Modifications and clarifications for the implementation of the Spalart-Allmaras turbulence model. Seventh International Conference on Computational Fluid Dynamics (ICCFD7) 1902, 2012.

[4] F. Bassi and S. Rebay. Numerical evaluation of two discontinuous Galerkin methods for the compressible Navier-Stokes, equations. *International Journal for Numerical Methods in Fluids*, 40:197–207, 2002.

[5] R. Becker and R. Rannacher. An optimal control approach to a posteriori error estimation in finite element methods. In A. Iserles, editor, *Acta Numerica*, pages 1–102. Cambridge University Press, 2001.

[6] Anca Belme, Alain Dervieux, and Frédéric Alauzet. Time accurate anisotropic goal-oriented mesh adaptation for unsteady flows. *Journal of Computational Physics*, 231(19):6323–6348, 2012.

[7] Patrick J. Blonigan, Steven A. Gomez, and Qiqi Wang. Least squares shadowing for sensitivity analysis of turbulent fluid flows. AIAA Paper 2014–1426, 2014.

[8] J.R. Cash. The integration of stiff initial value problems in ODEs using modified extended backward differentiation formulae. *Computers & mathematics with applications*, 9(5):645–657, 1983.

[9] Marco A. Ceze and Krzysztof J. Fidkowski. High-order output-based adaptive simulations of turbulent flow in two dimensions. AIAA Paper 2015–1532, 2015.

[10] Guodong Chen and Krzysztof J. Fidkowski. Discretization error control for constrained aerodynamic shape optimization. *Journal of Computational Physics*, 387:163–185, 2019.

[11] Bernardo Cockburn and Chi-Wang Shu. Runge-Kutta discontinuous Galerkin methods for convection-dominated problems. *Journal of Scientific Computing*, 16(3):173–261, 2001.

[12] Thomas D. Economon, Francisco Palacios, and Juan J. Alonso. Unsteady continuous adjoint approach for aerodynamic design on dynamic meshes. *AIAA Journal*, 53(9):2437–2453, 2015.

[13] Krzysztof J. Fidkowski. Output error estimation strategies for discontinuous Galerkin discretizations of unsteady convection-dominated flows. *International Journal for Numerical Methods in Engineering*, 88(12):1297–1322, 2011.

[14] Krzysztof J. Fidkowski. An output-based dynamic order refinement strategy for unsteady aerodynamics. AIAA Paper 2012-77, 2012.

[15] Krzysztof J. Fidkowski. A local sampling approach to anisotropic metric-based mesh optimization. AIAA Paper 2016–0835, 2016.

[16] Krzysztof J. Fidkowski. Output-based space-time mesh optimization for unsteady flows using continuous-in-time adjoints. *Journal of Computational Physics*, 341(15):258–277, July 2017.

[17] Krzysztof J. Fidkowski. Three-dimensional benchmark RANS computations using discontinuous finite elements on solution-adapted meshes. AIAA Paper 2018–1104, 2018.

[18] Krzysztof J. Fidkowski and David L. Darmofal. Review of output-based error estimation and mesh adaptation in computational fluid dynamics. *AIAA Journal*, 49(4):673–694, 2011.

[19] Krzysztof J. Fidkowski and Yuxing Luo. Output-based space-time mesh adaptation for the compressible Navier-Stokes equations. *Journal of Computational Physics*, 230:5753–5773, 2011.

[20] Krzysztof J. Fidkowski, Todd A. Oliver, James Lu, and David L. Darmofal. *p*-Multigrid solution of high–order discontinuous Galerkin discretizations of the compressible Navier-Stokes equations. *Journal of Computational Physics*, 207:92–113, 2005.

[21] Andreas Griewank and Andrea Walther. Revolve: An implementation of checkpointing for the reverse or adjoint mode of computational differentiation. *ACM Transactions on Mathematical Software*, 26(1):19–45, 2000.

[22] Joel Ho and Alastair West. Field inversion and machine learning for turbulence modelling applied to three-dimensional separated flows. AIAA Paper 2021–2903, 2021.

[23] Jonathan R. Holland, James D. Baeder, and Karthik Duraisamy. Towards integrated field inversion and machine learning with embedded neural networks for RANS modeling. AIAA Paper 2019-1884, 2019.

[24] Jr. J. E. Dennis and Jorge J. More. Quasi-newton methods, motivation and theory. *Society for Industrial and Applied Mathematics Review*, 19:359 – 372, 1977.

[25] Florian Jäckel. A closed-form correction for the Spalart-Allmaras turbulence model for separated flow. AIAA Paper 2022–0462, 2022.

[26] Claes Johnson. On computability and error control in CFD. *International Journal for Numerical Methods in Fludis*, 20:777–788, 1995.

[27] Steven M. Kast and Krzysztof J. Fidkowski. Output-based mesh adaptation for high order Navier-Stokes simulations on deformable domains. *Journal of Computational Physics*, 252(1):468–494, 2013.

[28] Joshua A. Krakos, Qiqi Wang, Steven R. Hall, and David L. Darmofal. Sensitivity analysis of limit cycle oscillations. *Journal of Computational Physics*, 231(8):3228–3245, 2012.

[29] Johan Larsson and Qiqi Wang. 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*, 372(2022):20130329, 2014.

[30] Dong C. Liu and Jorge Nocedal. On the limited memory BFGS method for large scale optimization. *Mathematical Programming*, 45:503 – 528, 1989.

[31] Y. Luo and Krzysztof J. Fidkowski. Output-based space time mesh adaptation for unsteady aerodynamics. AIAA Paper 2011-491, 2011.

[32] Karthik Mani and Dimitri J. Mavriplis. Error estimation and adaptation for functional outputs in time-dependent flow problems. *Journal of Computational Physics*, 229:415–440, 2010.

[33] Joaquim R. R. A. Martins and Andrew B. Lambe. Multidisciplinary design optimization: a survey of architectures. *AIAA Journal*, 51(9):2049–2075, 2013.

[34] D. Meidner and B. Vexler. Adaptive space-time finite element methods for parabolic optimization problems. *SIAM Journal on Control Optimization*, 46(1):116–142, 2007.

[35] Siva K. Nadarajah and Antony Jameson. A comparison of the continuous and discrete adjoint approach to automatic aerodynamic optimization. AIAA Paper 2000-0667, 2000.

[36] Marian Nemec and Michael J. Aftosmis. Error estimation and adaptive refinement for embedded-boundary Cartesian meshes. AIAA Paper 2007-4187, 2007.

[37] Angxiu Ni and Qiqi Wang. Sensitivity analysis on chaotic dynamical systems by non-intrusive least squares shadowing (NILSS). *Journal of Computational Physics*, 347:56–77, 2017.

[38] Angxiu Ni, Qiqi Wang, Pablo Fernández, and Chaitanya Talnikar. Sensitivity analysis on chaotic dynamical systems by finite difference non-intrusive least squares shadowing (FD-NILSS). *Journal of Computational Physics*, 394:615–631, 2019.

[39] Vivek Ojha, Krzysztof J. Fidkowski, and Carlos E. S. Cesnik. Adaptive mesh refinement for fluid-structure interaction simulations. AIAA Paper 2021–0731, 2021.

[40] Eric J. Parish and Karthik Duraisamy. A paradigm for data-driven predictive modeling using field inversion and machine learning. *Journal of Computational Physics*, 305:758–774, 2016.

[41] Olivier Pironneau. On optimum design in fluid mechanics. *Journal of Fluid Mechanics*, 64:97–110, 1974.

[42] W. Reed and T. Hill. Triangular mesh methods for the neutron transport equation. Los Alamos Scientific Laboratory Technical Report LA-UR-73-479, 1973.

[43] P.L. Roe. Approximate Riemann solvers, parameter vectors, and difference schemes. *Journal of Computational Physics*, 43:357–372, 1981.

[44] Benjamin A. Rothacker, Marco A. Ceze, and Krzysztof J. Fidkowski. Adjoint-based error estimation and mesh adaptation for problems with output constraints. AIAA Paper 2014–2576, 2014.

[45] Youcef Saad and Martin H. Schultz. GMRES: A generalized minimal residual algorithm for solving non-symmetric linear systems. *SIAM Journal on Scientific Computing*, 7(3):856–869, 1986.

[46] Yukiko S. Shimizu and Krzysztof J. Fidkowski. Output error estimation for chaotic flows. AIAA Paper 2016-3806, 2016.

[47] Anand Pratap Singh, Shivaji Medida, and Karthik Duraisamy. Machine-learning-augmented predictive modeling of turbulent separated flows over airfoils. *AIAA Journal*, 55(7):2215–2227, 2017.

[48] D. A. Venditti and D. L. Darmofal. Anisotropic grid adaptation for functional outputs: application to two-dimensional viscous flows. *Journal of Computational Physics*, 187(1):22–46, 2003.

[49] Qiqi Wang. Forward and adjoint sensitivity computation of chaotic dynamical systems. *Journal of Computational Physics*, 235:1–13, 2013.

[50] Qiqi Wang. Convergence of the least squares shadowing method for computing derivative of ergodic averages. *SIAM Journal on Numerical Analysis*, 52(1):156–170, 2014.

[51] Qiqi Wang, Rui Hu, and Patrick Blonigan. Least squares shadowing sensitivity analysis of chaotic limit cycle oscillations. *Journal of Computational Physics*, 267:210–224, 2014.

[52] Chongyang Yan, Haoran Li, Yufei Zhang, and Haixin Chen. Data-driven turbulence modeling in separated flows considering physical mechanism analysis, 2021.

[53] M. Yano and D.L. Darmofal. An optimization-based framework for anisotropic simplex mesh adaptation. *Journal of Computational Physics*, 231(22):7626–7649, 2012.

[54] Masayuki Yano. *An Optimization Framework for Adaptive Higher-Order Discretizations of Partial Differential Equations on Anisotropic Simplex Meshes*. PhD thesis, Massachusetts Institute of Technology, Cambridge, Massachusetts, 2012.