AIAA SciTech Forum
12-16 January 2026, Orlando, FL
AIAA SCITECH 2026 Forum

10.2514/6.2026-1212

Towards Hyper-Reduced Weighted POD for Large-Scale Aerodynamic Design Optimization

Sebastiaan P. C. van Schie*, John T. Hwang†
University of California San Diego, 9500 Gilman Dr., La Jolla, CA 92093

The aim of this work is to present the first steps of our research to make high-fidelity inviscid compressible flow models computationally tractable for large-scale preliminary aircraft design optimization problems. We take a novel weighted Proper Orthogonal Decomposition (POD) approach and apply it to speed up a Discontinuous Galerkin-based compressible Euler model in an airfoil shape optimization problem. We show how the weighted POD approach improves upon a standard POD approach, especially when predicting drag coefficients, while having lower wall times than the standard POD approach. We intend to extend the current approach with hyper-reduction, in order to obtain further speedups, and switch to gradient-based optimization methods in order to enable more efficient solving of large-scale optimization problems.

I. Introduction

Consistent with Moore's law, the cost of computing power has increased exponentially over the last couple of decades. The desired scope and fidelity of Partial Differential Equation (PDE)-constrained design optimization problems has expanded at a similar pace in order to take up the available computational resources. For example, transonic aircraft optimization studies carried out 25 years ago [1] and now [2] both use $\mathcal{O}\left(10^3\right) - \mathcal{O}\left(10^4\right)$ CPU core hours, despite the increases in CPU speeds over the past few decades. In this work we aim to reduce the computational cost of large-scale PDE-constrained optimization problems by applying model reduction, in order to make large-scale high-fidelity optimization feasible in the early stages of the design process. The reduced model produces approximate solutions to the PDE constraints that progress the optimization process for a fraction of the cost of computing solutions to the original PDE constraint model.

We focus here on intrusive model reduction of a finite element model of the steady compressible Euler equations. The compressible Euler equations have been used for design optimization in a range of studies [1, 3–6]. The application of model reduction techniques to compressible fluid flows was reviewed by Mendonça et al. in [7]. Various studies have investigated the stability of reduced compressible flow models. A small selection is [8–10], wherein particular attention is given to the stability of Proper Orthogonal Decomposition (POD) approaches. In this work we use the weighted POD approach that was proposed in [11], and compare it to a POD approach with a global basis. The resulting parametric reduced model is used to carry out gradient-free shape optimization.

The novelty of this work is present in its model reduction approach. We apply the novel weighted POD approach proposed in [11] to a steady compressible Euler model, and leverage it to speed up optimization problems. This presents the first few steps in our work towards demonstrating efficient hyper-reduced POD methods for large-scale aerodynamic design optimization.

The rest of this work is structured as follows. First, in Section II, we give a brief overview of the computational models that are used. This includes the governing Discontinuous Galerkin (DG) Euler equation model, stabilization terms and the mesh morphing approach necessary for shape optimization. Second, we give a brief overview of the model reduction approaches that are used in Section III. Third, in Section IV, we cover how the reduced model interacts with the PDE-constrained optimization problem, without the need to modify the optimization algorithm. Fourth, in Section V we demonstrate the accuracy of the DG model by comparing it to experimental results. Following this we carry out airfoil shape optimization with the weighted POD approach and compare its performance to a baseline POD model. Lastly, in Section VI we recap the preceding sections and give a perspective on the next steps in this research.

*Ph.D. Candidate, Department of Mechanical and Aerospace Engineering, AIAA student member
†Associate Professor, Department of Mechanical and Aerospace Engineering, AIAA senior member

II. Computational Models

In this section we cover the computational model that is used to solve aerodynamic design optimization problems. This consists of two parts: First, in Section II.A, we cover the numerical compressible Euler model. To be able to do shape optimization we also require a mesh motion model, which is described in Section II.B.

A. Compressible Euler Model

We use a Discontinuous Galerkin (DG) discretization of the compressible Euler equations, through the `dolfin_dg` package [12], implemented in FEniCS [13]. We opt for a DG-based discretization in order to align the numerical model choice with the hyperbolic character of the compressible Euler equations. In their steady form, these can be formulated as

$$\nabla \cdot \mathcal{F}(\boldsymbol{y}) = \boldsymbol{0} \quad \text{in } \Omega, \tag{1}$$

where $\Omega \subset \mathbb{R}^d$ is a spatial domain, with $d \in \{2, 3\}$. The solution vector $\boldsymbol{y} : \Omega \to \mathbb{R}^+ \times \mathbb{R}^d \times \mathbb{R}^+$ consists of

$$\boldsymbol{y} = \begin{bmatrix} \rho \\ \rho \boldsymbol{u} \\ \rho E \end{bmatrix}, \tag{2}$$

with ρ the fluid density, $\rho\boldsymbol{u}$ the fluid momentum, and ρE the total energy density. In (1), $\mathcal{F}$ is a flux function, formulated for the steady compressible Euler equations as

$$\mathcal{F}(\boldsymbol{y}) = \begin{pmatrix} \rho \boldsymbol{u} \\ \rho \boldsymbol{u} \otimes \boldsymbol{u} + p\boldsymbol{I} \\ (\rho E + p)\boldsymbol{u} \end{pmatrix}. \tag{3}$$

In this way (1) encodes conservation of mass, momentum and energy. To arrive at a closed system with a unique solution we need to add an equation of state, which defines the fluid pressure p as a function of the other field variables. For this we use

$$p = (\gamma - 1)\rho \left(E - \frac{1}{2} ||\boldsymbol{u}||^2 \right), \tag{4}$$

where γ is the ratio of specific heats; we use $\gamma = 1.4$ in the rest of this work.

As is usual for finite element methods, we take the inner product of (1) with a vector-valued test function $\boldsymbol{v}$ and integrate the result over Ω,

$$\int_\Omega (\nabla \cdot \mathcal{F}(\boldsymbol{y})) \cdot \boldsymbol{v} \; d\Omega = 0. \tag{5}$$

DG methods then use integration by parts, in order to arrive at

$$-\int_\Omega \mathcal{F}(\boldsymbol{y}) \colon \nabla \boldsymbol{v} \; d\Omega + \int_\Gamma (\boldsymbol{n} \cdot \mathcal{F}(\boldsymbol{y})) \cdot \boldsymbol{v} \; d\Gamma = 0, \tag{6}$$

with $\Gamma = \partial\Omega$ and $\boldsymbol{n}$ the outward-pointing normal vector on Γ. We discretize (6) by covering Ω with n_k cells $\{\Omega_k\}_{k=1}^{n_k}$, such that $\Omega = \bigcup_{k=1}^{n_k} \overline{\Omega}_k$ and $\Omega_i \cap \Omega_j = \emptyset$ for $i \neq j$. We designate the boundary overlap of cells i and j as $\overline{\Omega}_i \cap \overline{\Omega}_j = \Gamma_{ij}$. On the set of cells $\{\Omega_k\}_{k=1}^{n_k}$ we define a finite-dimensional function basis. For finite element methods these often consist of overlapping locally-supported polynomial functions; in other words, polynomial functions that are nonzero on only a handful of neighboring cells. What separates DG methods from other finite element approaches, is that each cell possesses its own function basis, independent from the bases on neighboring cells. For a cell Ω_k, (6) then becomes

$$-\int_{\Omega_k} \mathcal{F}(\boldsymbol{y}_k) \colon \nabla \boldsymbol{v}_k \; d\Omega_k + \sum_j \int_{\Gamma_{kj}} \left(\boldsymbol{n}_{kj} \cdot \mathcal{F}(\boldsymbol{y}_k) \right) \cdot \boldsymbol{v}_k \; d\Gamma_{kj} = 0, \tag{7}$$

with j the index that loops over all neighboring cells of Ω_k. However, this leaves ambiguous what the value of $\mathcal{F}(\boldsymbol{y}_k)$ on Γ_{kj} is. Conservation properties require that the flux $\mathcal{F}$ on Γ_{kj} should be identical and opposite on either side of Γ_{kj}, such that the flux from cell k to j is equal and opposite to the flux from cell j to k. Since $\boldsymbol{y}_k$ and $\boldsymbol{y}_j$ are expressed in separate function bases, they can have different values on Γ_{kj}. To achieve flux consistency, we replace $\boldsymbol{n}_{kj} \cdot \mathcal{F}(\boldsymbol{y}_k)$ on Γ_{kj} by a flux function $\mathbb{F}$ that depends on both $\boldsymbol{y}_k$ and $\boldsymbol{y}_j$. Equation (7) then becomes

$$-\int_{\Omega_k} \mathcal{F}(\boldsymbol{y}_k) : \nabla \boldsymbol{v}_k \, d\Omega_k + \sum_j \int_{\Gamma_{kj}} \mathbb{F}(\boldsymbol{y}_k, \boldsymbol{y}_j) \cdot \boldsymbol{v}_k \, d\Gamma_{kj} = 0. \tag{8}$$

Through the flux function we can thus make the solutions on neighboring cells depend on one another, even though each cell is covered by independent bases.

As mentioned earlier, flux functions are used to enforce the conservation of physical quantities such as mass and momentum between neighboring cells, similar to finite volume methods. When using piecewise-constant function bases on all cells, DG methods reduce to finite volume methods. The book by Hesthaven and Warburton [14] gives an in-depth treatment of DG methods. Houston and Sime [12] define the specific DG formulation that is used in this work. We use the Lax-Friedrichs flux function, defined as

$$\mathbb{F}_{kj}^{LF} = \frac{1}{2}\left(\mathcal{F}(\boldsymbol{y}^+) \cdot \boldsymbol{n}_{kj} + \mathcal{F}(\boldsymbol{y}^-) \cdot \boldsymbol{n}_{kj}\right) + \frac{\alpha}{2}\left(\boldsymbol{y}^+ - \boldsymbol{y}^-\right), \tag{9}$$

where $\boldsymbol{y}^+$ and $\boldsymbol{y}^-$ are the solution vectors on either side of the boundary Γ_{kj}; which of $\boldsymbol{y}_k$ and $\boldsymbol{y}_j$ correspond to the plus or minus superscript depends on the orientation of Γ_{kj}. Moreover, $\alpha \in \mathbb{R}$ is a local stabilization parameter. In [12] the procedure for computing α is discussed.

In order to make the DG formulation from `dolfin_dg` more robust for use in optimization problems, we add an artificial viscosity term,

$$\int_\Omega \eta \nabla \boldsymbol{y} \cdot \nabla \boldsymbol{v} \, d\Omega, \tag{10}$$

where

$$\eta = h^{2-\beta} \left|\nabla \cdot \mathcal{F}(\boldsymbol{y})\right| I. \tag{11}$$

Here h is the local mesh cell diameter, $0 < \beta < 1/2$, and I is the identity matrix. This is the same stabilization approach as used by Hartmann and Houston in [15].

Discretizing the stabilized DG formulation leaves us with residual vector $\boldsymbol{R}(\boldsymbol{y}, \boldsymbol{\mu})$, which describes how well the solution $\boldsymbol{y}$ satisfies the discretized model which also depends on parameter vector $\boldsymbol{\mu} \in \mathbb{R}^p$. Valid numerical solutions correspond to $\boldsymbol{R}(\boldsymbol{y}, \boldsymbol{\mu}) = \boldsymbol{0}$. We compute such solutions through the Newton-Raphson method,

$$\frac{\partial \boldsymbol{R}(\boldsymbol{y}^i, \boldsymbol{\mu})}{\partial \boldsymbol{y}^i} \Delta \boldsymbol{y}^{i+1} = -\boldsymbol{R}(\boldsymbol{y}^i, \boldsymbol{\mu}). \tag{12}$$

During each iteration we compute a solution update $\Delta \boldsymbol{y}^{i+1}$, such that

$$\boldsymbol{y}^{i+1} = \Delta \boldsymbol{y}^{i+1} + \boldsymbol{y}^i. \tag{13}$$

Convergence can be judged by computing the norm of $\boldsymbol{R}(\boldsymbol{y}^i, \boldsymbol{\mu})$. For $\boldsymbol{y}^i \in \mathbb{R}^N$, the Jacobian matrix $\frac{\partial \boldsymbol{R}(\boldsymbol{y}^i, \boldsymbol{\mu})}{\partial \boldsymbol{y}^i} \in \mathbb{R}^{N \times N}$. For large N solving (12) can come at a significant cost. Computing a solution $\boldsymbol{y}$ for which $\boldsymbol{R}(\boldsymbol{y}, \boldsymbol{\mu}) \approx \boldsymbol{0}$ requires solving (12) multiple times.

B. Mesh Morphing Model

A major intended application for the reduced model developed in this work is aerodynamic shape design, handled through Free-Form Deformation (FFD). In future work we intend to use gradient-based optimization methods, which require us to keep the mesh topology consistent during any optimization process. As such, any object shape changes induced by the optimization process need to be applied to the mesh in a way which ensures that the resulting deformed mesh is numerically well-conditioned and of sufficient quality. With this in mind we use a mesh morphing approach in this work.

The object shape changes amount to mesh boundary deformations. We are interested in propagating these deformations throughout the mesh in a smooth way. To achieve this we solve the linear elasticity problem

$$\int_{\Omega} \boldsymbol{\sigma}(\boldsymbol{w}) : \boldsymbol{\epsilon}(\boldsymbol{d}) \, d\Omega = 0 \tag{14}$$

for displacement $\boldsymbol{d}$ and with test function $\boldsymbol{w}$, where

$$\begin{aligned} \boldsymbol{\epsilon}(\boldsymbol{d}) &= \frac{1}{2}\left(\nabla \boldsymbol{d} + (\nabla \boldsymbol{d})^T\right) \\ \boldsymbol{\sigma}(\boldsymbol{w}) &= \lambda \, \mathrm{tr}\,(\boldsymbol{\epsilon}(\boldsymbol{w}))\, I + 2\mu \boldsymbol{\epsilon}(\boldsymbol{w}). \end{aligned} \tag{15}$$

Here $\boldsymbol{\sigma}$ and $\boldsymbol{\epsilon}$ are the stress and symmetric strain tensors respectively. The Lamé parameters λ and μ depend on the Young's modulus E and Poisson's ratio ν, for which we use $E = 1$ and $\nu = 0.4$. This dependence is

$$\begin{aligned} \mu &= \frac{E}{2(1+\nu)} \\ \lambda &= \frac{E\nu}{(1+\nu)(1-2\nu)}. \end{aligned} \tag{16}$$

Equation (14) is discretized in a (non-DG) linear function basis. As boundary conditions for (14) we use zero deformation Dirichlet conditions on the outer boundaries of Ω, and impose the deformations that arise from FFD on the object boundaries in the middle of the domain. The deformations that are computed with the mesh morphing model can be added directly to the coordinates of the nodes of the baseline mesh, which then results in a mesh that is smoothly deformed by FFD shape changes during the optimization process.

III. Model Reduction

As mentioned previously, obtaining valid solutions to the steady compressible Euler equations requires solving (12) multiple times. Due to the potentially large size of the matrix and vectors in (12), computing the solution updates $\Delta \boldsymbol{y}^{i+1}$ can present significant computational costs. These costs are divided over two main steps:

1) Assembly and construction of $\frac{\partial \boldsymbol{R}(\boldsymbol{y}^i, \boldsymbol{\mu})}{\partial \boldsymbol{y}^i}$ and $\boldsymbol{R}(\boldsymbol{y}^i, \boldsymbol{\mu})$.
2) Computing the solution $\Delta \boldsymbol{y}^{i+1}$ to the assembled system (12).

In this work we focus on reducing the cost of the latter of these two steps.

First, we cover the standard POD approach in Section III.A, which addresses the cost of computing solutions to (12). Following that we cover in Section III.B an extension of the standard POD approach, which lets one take into account any parametric structure underpinning the data sets that are used in the POD process, with the aim of creating more accurate POD models.

A. Proper Orthogonal Decomposition

Proper Orthogonal Decomposition aims to speed up the process of computing solutions to (12) by instead computing an approximate, n-dimensional solution with $n \ll N$. Since the cost of solving (12) scales with the size of the linear system, low-dimensional solutions can be computed more efficiently. In this work we use the POD-Galerkin method, in which we project (12) to an n-dimensional subspace by left-multiplying with $\boldsymbol{\Phi}^T \in \mathbb{R}^{n\times N}$ and projecting $\boldsymbol{y}^i$ to the same n-dimensional subspace,

$$\begin{aligned} \boldsymbol{y}^i &= \boldsymbol{\Phi}\boldsymbol{y}_r^i \\ \Delta \boldsymbol{y}^{i+1} &= \boldsymbol{\Phi}\Delta \boldsymbol{y}_r^{i+1}, \end{aligned} \tag{17}$$

with $\boldsymbol{y}_r^i, \Delta \boldsymbol{y}_r^{i+1} \in \mathbb{R}^n$. The reduced Newton-Raphson iterations are then

$$\boldsymbol{\Phi}^T \frac{\partial \boldsymbol{R}(\boldsymbol{y}^i, \boldsymbol{\mu})}{\partial \boldsymbol{y}^i} \boldsymbol{\Phi}\Delta \boldsymbol{y}_r^{i+1} = -\boldsymbol{\Phi}^T \boldsymbol{R}(\boldsymbol{y}^i, \boldsymbol{\mu}). \tag{18}$$

We refer to $\boldsymbol{\Phi}$ as the reduced basis projection matrix. We want to construct $\boldsymbol{\Phi}$ such that $\boldsymbol{y}_r^{i+1}$ is close to the full-dimensional solution $\boldsymbol{y}^{i+1}$ of (12).

Suppose we have available a set of known solutions to (12), corresponding to different values of a parameter vector $\boldsymbol{\mu}$ in the parameter space $\mathbb{P} \subset \mathbb{R}^p$. We designate this set of solutions as $\left\{\boldsymbol{y}^j(\boldsymbol{\mu}^j)\right\}_{j=1}^s$. We gather these solutions in the snapshot matrix

$$\boldsymbol{X} = \begin{bmatrix} \boldsymbol{y}^1(\boldsymbol{\mu}^1) & \boldsymbol{y}^2(\boldsymbol{\mu}^2) & \dots & \boldsymbol{y}^s(\boldsymbol{\mu}^s) \end{bmatrix} \in \mathbb{R}^{N \times s}. \tag{19}$$

When given a new parameter vector $\hat{\boldsymbol{\mu}} \notin \left\{\boldsymbol{\mu}^j\right\}_{j=1}^s$, we want to compute the reduced solution that best approximates $\boldsymbol{y}(\hat{\boldsymbol{\mu}})$. As mentioned before, the accuracy of the approximation depends on $\boldsymbol{\Phi}$. To construct $\boldsymbol{\Phi}$ we first compute the Singular Value Decomposition (SVD) of $\boldsymbol{X}$,

$$\boldsymbol{X} = \boldsymbol{\Psi}\boldsymbol{\Sigma}\boldsymbol{\Pi}^T = \underbrace{\begin{bmatrix} \boldsymbol{\psi}_1 & \boldsymbol{\psi}_2 & \dots \end{bmatrix}}_{\in\mathbb{R}^{N\times s}} \underbrace{\begin{bmatrix} \sigma_1 & & \\ & \sigma_2 & \\ & & \ddots \end{bmatrix}}_{\in\mathbb{R}^{s\times s}} \underbrace{\begin{bmatrix} \boldsymbol{\pi}_1^T \\ \boldsymbol{\pi}_2^T \\ \vdots \end{bmatrix}}_{\in\mathbb{R}^{s\times s}}, \tag{20}$$

and take as reduced basis $\boldsymbol{\Phi}$ the first n columns of $\boldsymbol{\Psi}$,

$$\boldsymbol{\Phi} = \begin{bmatrix} \boldsymbol{\psi}_1 & \boldsymbol{\psi}_2 & \dots & \boldsymbol{\psi}_n \end{bmatrix}. \tag{21}$$

This makes $\boldsymbol{\Phi}$ independent of $\hat{\boldsymbol{\mu}}$, and takes into account solution information that is obtained all throughout $\mathbb{P}$. We refer to this choice of $\boldsymbol{\Phi}$ as the global POD basis.

B. Snapshot Weighting for Proper Orthogonal Decomposition

In [11] we proposed using a weighted POD approach to efficiently construct accurate reduced bases for parametric models. This is especially useful for nonlinear models, whose solutions can drastically change for different parameter inputs. We showed in [11] that the $\mathbb{X}$-norm error between $\boldsymbol{y}(\hat{\boldsymbol{\mu}})$ and $\boldsymbol{y}_r(\hat{\boldsymbol{\mu}})$ can be bounded from above by

$$\|\boldsymbol{y}(\hat{\boldsymbol{\mu}}) - \boldsymbol{y}_r(\hat{\boldsymbol{\mu}})\|_{\mathbb{X}}^2 \leq 3s \sum_{i=n+1}^{s} \sigma_i^2 + 3sp \sum_{j=1}^{s} \omega_j^2 \left[\left\| \hat{\boldsymbol{\mu}} - \boldsymbol{\mu}^j + \rho_j^1 \mathbf{1} \right\|_{\boldsymbol{C}_j^1}^2 + \left\| \hat{\boldsymbol{\mu}} - \boldsymbol{\mu}^j + \rho_j^3 \mathbf{1} \right\|_{\boldsymbol{C}_j^3}^2 \right], \tag{22}$$

with ρ_j^1, ρ_j^3 radii of open balls centered around $\boldsymbol{\mu}^j$ for a regularization step, and $\boldsymbol{C}_j^1, \boldsymbol{C}_j^3$ indicating a specifically weighted norm that bounds the $\mathbb{X}$-norm from above. More details and a corresponding proof of a more general, time-dependent version of (22) are given in [11]. To summarize the conclusions that can be drawn from (22): The POD-Galerkin approximation error expressed in the $\mathbb{X}$-norm is bounded from above by the sum of:

1) The singular values in $\boldsymbol{\Sigma}$ that do not correspond to any of the columns of $\boldsymbol{\Psi}$ that are used in the reduced basis projection matrix $\boldsymbol{\Phi}$.
2) A multiple of a sum of regularized distance norms between the new parameter vector $\hat{\boldsymbol{\mu}}$ and each of the snapshot parameter vectors $\boldsymbol{\mu}^j$, multiplied with snapshot weights ω_j^2.

While we do not have direct control over the sum of neglected singular values, we do have control over the snapshot weights ω_j. As is implied by (22), choosing larger snapshot weights for snapshots that are located closer to $\hat{\boldsymbol{\mu}}$ allows one to minimize the upper bound. Note that in the proof of (22) in [11], $\left\{\omega_j\right\}_{j=1}^s$ forms a partition of unity. The global POD approach that was covered in Section III.A is obtained when using $\omega_j = 1/s \; \forall \; j \in \{1, 2, \dots, s\}$. For any other combination of snapshot weights, the weighted POD basis can be obtained by right-multiplying the SVD of $\boldsymbol{X}$ with the diagonal weight matrix $\boldsymbol{W}$, such that

$$\boldsymbol{X}\boldsymbol{W} = \boldsymbol{\Psi}\boldsymbol{\Sigma}\boldsymbol{\Pi}^T\boldsymbol{W} = \underbrace{\begin{bmatrix} \boldsymbol{\psi}_1 & \boldsymbol{\psi}_2 & \dots \end{bmatrix}}_{\in\mathbb{R}^{N\times s}} \underbrace{\begin{bmatrix} \sigma_1 & & \\ & \sigma_2 & \\ & & \ddots \end{bmatrix}}_{\in\mathbb{R}^{s\times s}} \underbrace{\begin{bmatrix} \boldsymbol{\pi}_1^T \\ \boldsymbol{\pi}_2^T \\ \vdots \end{bmatrix}}_{\in\mathbb{R}^{s\times s}} \underbrace{\begin{bmatrix} \omega_1 & & \\ & \omega_2 & \\ & & \ddots \end{bmatrix}}_{\in\mathbb{R}^{s\times s}}. \tag{23}$$

The SVD of the weighted snapshot matrix $\boldsymbol{X}\boldsymbol{W}$ can be obtained cheaply once the SVD of $\boldsymbol{X}$ is known. We form the matrix $\boldsymbol{D} = \boldsymbol{\Sigma}\boldsymbol{\Pi}^T\boldsymbol{W} \in \mathbb{R}^{s\times s}$ and compute its SVD,

$$\boldsymbol{D} = \boldsymbol{\Sigma}\boldsymbol{\Pi}^T\boldsymbol{W} = \boldsymbol{\Psi}_{\boldsymbol{D}}\boldsymbol{\Sigma}_{\boldsymbol{D}}\boldsymbol{\Pi}_{\boldsymbol{D}}^T. \tag{24}$$

Then, the weighted snapshot matrix $\boldsymbol{XW}$ possesses the following SVD,

$$\begin{aligned} \boldsymbol{\Psi}_{\boldsymbol{XW}} &= \boldsymbol{\Psi}\boldsymbol{\Psi}_{\boldsymbol{D}} \\ \boldsymbol{\Sigma}_{\boldsymbol{XW}} &= \boldsymbol{\Sigma}_{\boldsymbol{D}} \\ \boldsymbol{\Pi}_{\boldsymbol{XW}} &= \boldsymbol{\Pi}_{\boldsymbol{D}}. \end{aligned} \tag{25}$$

The weighted POD basis $\boldsymbol{\Phi}_{\boldsymbol{W}}$ is then equal to the first n columns of $\boldsymbol{\Psi}_{\boldsymbol{XW}}$.

This leaves open the question of how the weights ω_j should be determined. We do this in the same way as was proposed in [11]. First, we compute the distance $\left\|\hat{\boldsymbol{\mu}} - \boldsymbol{\mu}^j\right\|_2$ between the new parameter vector $\hat{\boldsymbol{\mu}}$ and each stored snapshot. Let δ_{min} and δ_{max} denote the minimum and maximum snapshot distances. We then define the cutoff distance as a linear combination of these distances,

$$\hat{\delta} = c\delta_{min} + (1-c)\delta_{max}, \tag{26}$$

with $c \in (0, 1]$. As is explained in more detail in [11], any monotonically non-increasing function can be used to then assign a weight $\omega_j > 0$ to any snapshot with a distance to $\hat{\boldsymbol{\mu}}$ that is smaller than $\hat{\delta}$. In the rest of this work we use $c = 0.7$ wherever the weighted POD basis is used.

IV. Design Optimization with POD

The general form of a PDE-constrained optimization problem is

$$\begin{aligned} \text{minimize} \quad & f(\boldsymbol{y}, \boldsymbol{\mu}) \in \mathbb{R} \\ \text{with respect to} \quad & \boldsymbol{y} \in \mathbb{R}^N \\ & \boldsymbol{\mu} \in \mathbb{P} \subseteq \mathbb{R}^p \\ \text{subject to} \quad & \boldsymbol{R}(\boldsymbol{y}, \boldsymbol{\mu}) = \boldsymbol{0} \\ & \boldsymbol{c}(\boldsymbol{y}, \boldsymbol{\mu}) \geq \boldsymbol{0}, \end{aligned} \tag{27}$$

with f a scalar-valued objective function and $\boldsymbol{c}$ a vector of (in)equality constraints. When using POD we do not exactly satisfy the PDE residual constraint $\boldsymbol{R}(\boldsymbol{y}, \boldsymbol{\mu}) = \boldsymbol{0}$. Instead, we compute reduced solutions $\boldsymbol{y}_r$ such that

$$\boldsymbol{\Phi}^T \boldsymbol{R}(\boldsymbol{\Phi}\boldsymbol{y}_r, \boldsymbol{\mu}) = \boldsymbol{0}. \tag{28}$$

Consequently, $\boldsymbol{R}(\boldsymbol{\Phi}\boldsymbol{y}_r, \boldsymbol{\mu}) > \boldsymbol{0}$. The non-zero residual correlates with the approximation error of the reduced model, and can be used to estimate this error [16]. As such we use a residual-based criterion to judge whether a POD solution should be accepted or rejected during the optimization process. Specifically, POD solutions should meet

$$\|\boldsymbol{R}(\boldsymbol{\Phi}\boldsymbol{y}_r, \boldsymbol{\mu})\|_2 < 5 \cdot 10^{-4}. \tag{29}$$

POD solutions $\boldsymbol{y}_r$ that do not meet this criterion are rejected, and the Full-Order Model (FOM) is then used to generate a solution that meets $\boldsymbol{R}(\boldsymbol{y}, \boldsymbol{\mu}) = \boldsymbol{0}$. This is comparable to the trust-region-based approach of Zahr and Farhat [17].

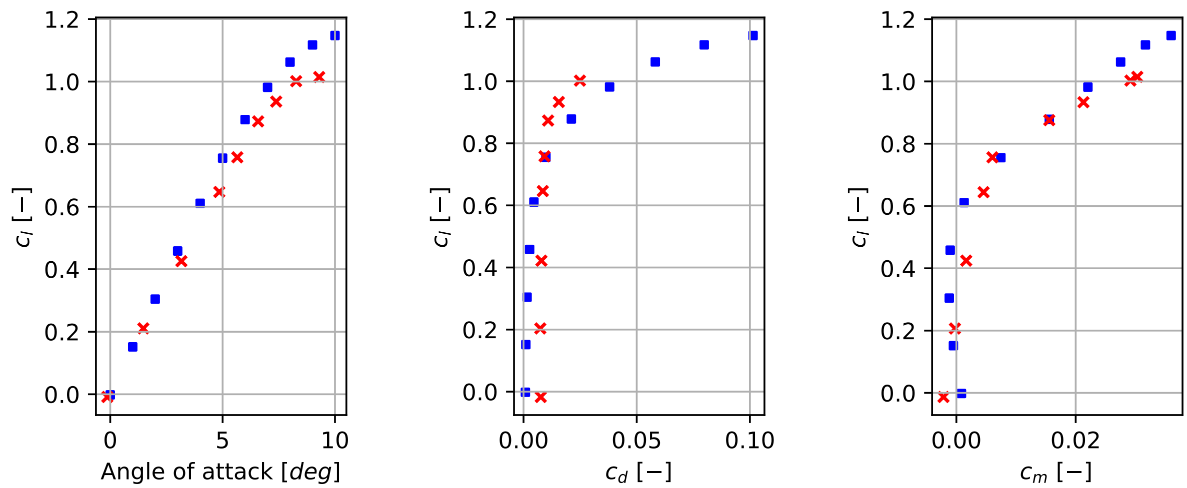
In the rest of this work we use the gradient-free COBYLA optimizer. As its name implies, the Constrained Optimization BY Linear Approximation (COBYLA) optimizer uses model evaluations to create linear approximations of the objective function and constraint derivatives. The DG model and mesh morphing model used here during optimization are interfaced through the Computational Systems Design Language [18]. We handle optimization with COBYLA by using the modOpt optimization package [19].

V. Numerical Experiments

We present here two sets of results. First, in Section V.A, we compare the predictions of the DG model with experimental data from literature. Then, in Section V.B, we carry out airfoil shape optimizations with the model reduction approaches that are covered in Section III.

A. Euler Model Validation

The data given in [20] is used to determine the accuracy of the DG model for compressible subsonic flows. This data includes lift, drag, and moment coefficients for a range of angles of attack of a NACA0012 airfoil at $M = 0.6$ and $Re = 9 \cdot 10^6$, as well as the pressure coefficient distribution. We emphasize that the DG model results do not involve any model reduction. Figure 1 shows the aforementioned force and moment coefficients, graphed over a range of angles of attack.



(a) Lift coefficient vs angle of attack **(b) Drag coefficient vs lift coefficient** **(c) Moment coefficient vs lift coefficient**

Fig. 1 Force and moment coefficients for a range of angles of attack of a NACA0012 airfoil at $M = 0.6$, $Re = 9 \cdot 10^6$. Numerical results indicated with blue blocks, experimental data from [20] indicated with red crosses

As can be seen in Figure 1a, the numerical and experimental lift coefficients line up well. Since the Euler equations are inviscid, it does not model flow separation. This explains why the numerical lift coefficients at higher angles of attack are larger than the experimental ones. Moreover, a shockwave develops on the upper airfoil around 4 degrees angle of attack. The DG model predicts the position of this shock to be further aft than the experimental data, likely due to the lack of boundary layer interactions. As a result, the DG model slightly over-predicts the resulting post-shock lift coefficient. The inviscid nature of the Euler equations is also responsible for the drag coefficient under-predictions shown in Figure 1b at $c_l < 0.6$: No viscous drag is present, and therefore the numerical model only resolves the lift-induced drag and the pressure drag components. At small lift coefficients the viscous drag is the dominant drag component, but at higher lift coefficients the other two aforementioned drag components start to dominate. Figure 1c shows that the experimental and numerical moment coefficients are close to one another for $c_l < 1$. For different lift coefficient values the DG model either slightly under- or over-predicts the moment coefficient, although the error relative to the experimental data is small.

Lastly, we look at the pressure coefficient distributions at an angle of attack of 3.16 degrees. These are shown in Figure 2. As can be seen, the experimental and numerical c_p distributions agree well with one another for most of the chord length. Near the trailing edge a small discrepancy between the two exists, likely due to the lack of viscosity in the numerical model. Moreover, the numerical model produces oscillatory pressure results near the area with the lowest pressure coefficients, in the area of initial shock onset. While these oscillations affect the local solution quality, they do not seem to significantly affect the predicted force and moment coefficients, as is evidenced by Figure 1.

B. Airfoil Shape Optimization

With the accuracy of the baseline DG model established, we turn to shape optimization. Starting from the NACA0012 airfoil, we aim to minimize its drag coefficient at $M = 0.6$ and an angle of attack of 3 degrees by modifying its shape, while requiring that $c_l \geq 0.4$. As covered in Section II.B, we use FFD to modify the airfoil's shape. We cover the airfoil with a Cartesian grid of FFD control points, 10 in chordwise direction and 3 in the vertical direction. The set of design variables consists of the vertical displacements of the control points. We constrain the control point at the upper side of

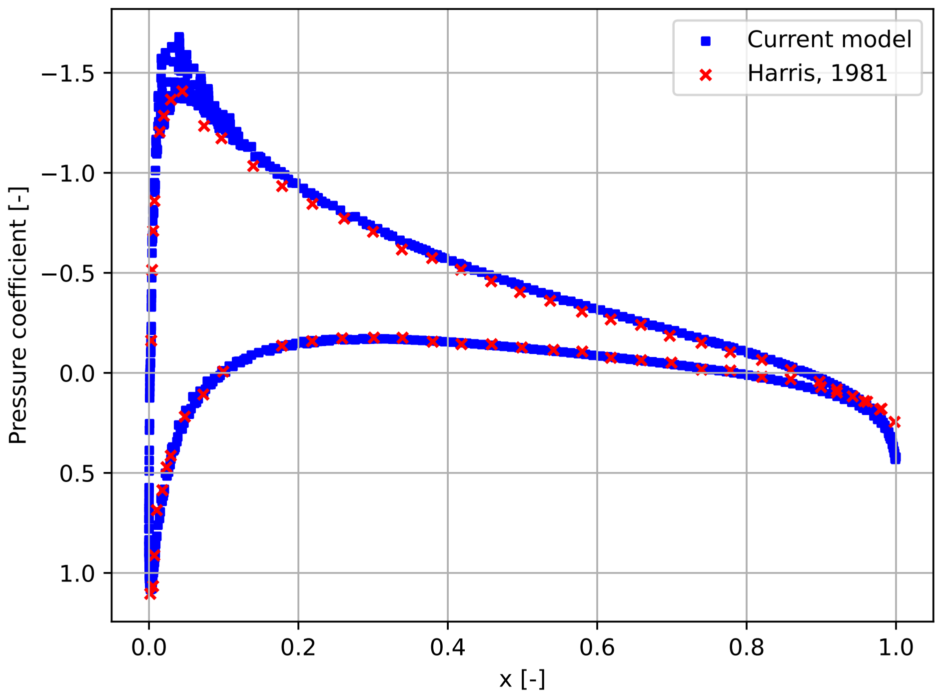


Fig. 2 Numerical and experimental pressure coefficient distributions around NACA0012 airfoil at $M = 0.6$, $\alpha = 3.16$ degrees

Table 1 Optimal designs obtained with global POD

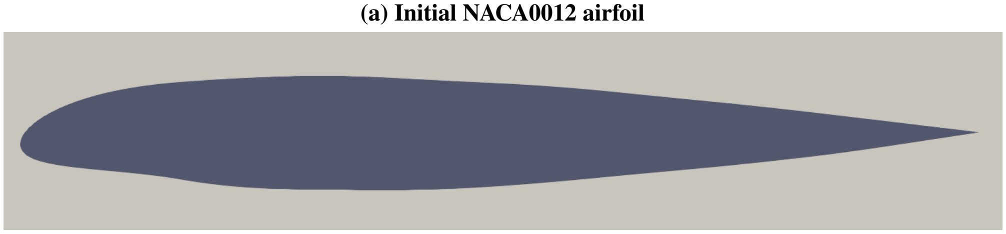
	ROM		FOM				
n	c_d	c_l	c_d	c_l	Model calls	ROM solves	FOM solves
10	0.00140	0.48585	0.00208	0.48887	147	104	43
20	0.00029	0.51103	0.00201	0.48267	231	181	50
30	0.00195	0.35479	0.00166	0.35864	146	111	35

Table 2 Optimal designs obtained with weighted POD

	ROM		FOM				
n	c_d	c_l	c_d	c_l	Model calls	ROM solves	FOM solves
10	0.00209	0.40276	0.00186	0.39900	125	79	46
20	0.00169	0.39588	0.00187	0.39752	126	88	38
30	0.00208	0.38795	0.00187	0.39756	125	87	38

the leading edge to remain in place, in order to rule out any rigid body modes. In order to avoid neighboring control points from crossing one another we limit their vertical displacements to the range $[-0.04, 0.04]$. The mesh for the DG simulations was obtained by using a dual-weighted residual-based refinement approach on the initial NACA0012 airfoil, adapted from the `dolfin_dg` repository*. This results in a mesh with about $178,000$ degrees of freedom for linear DG elements. Figure 3 shows the initial and optimized airfoil shapes when only using the FOM. The optimized drag coefficient is $c_d = 0.0012$. As can be seen in Figure 3, the leading edge of the airfoil is moved downward, and the airfoil surface is pinched near the high pressure side of the leading edge. Moreover, the trailing edge is curved slightly upwards in the optimized airfoil design.

(a) Initial NACA0012 airfoil

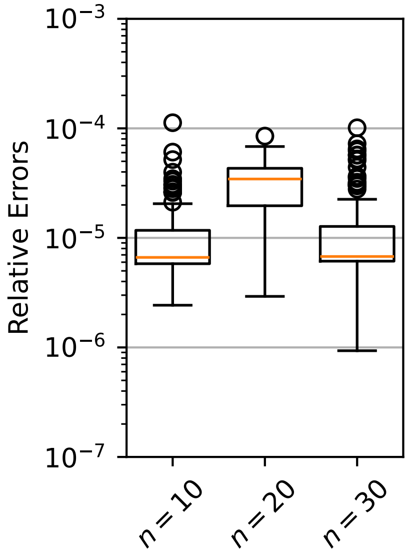


(b) FOM-optimized airfoil shape

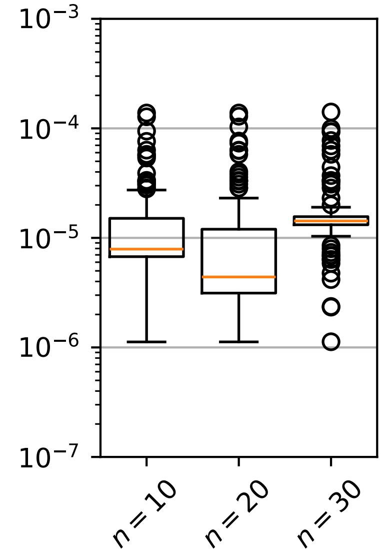
Fig. 3 Initial and optimized airfoil shape computed with the FOM

We use the global and weighted POD approaches as covered in Section III to aid in the optimization process, as discussed in Section IV. We test reduced basis sizes of $n = 10, 20, 30$. First we look at the performance of the optimized airfoil shapes that are found in this way, and compare their performance as predicted by the ROMs to their actual performance. The results are shown in Table 1 for the global POD basis, and Table 2 for the weighted POD basis. First we look at the accuracy of the c_d predictions. These are significantly less accurate for the global POD approach than for the weighted POD approach. The same is true for the lift coefficients. Recall that we impose a constraint of $c_l \geq 0.4$. Due to the lift-induced drag component we expect this constraint to be tight for any optimal design. Despite that, the

*`https://github.com/nate-sime/dolfin_dg`



(a) Global POD



(b) Weighted POD

Fig. 4 Relative L_2 errors, obtained with the global and weighted POD approaches

global POD approach results in c_l values that are either significantly larger or smaller than 0.4. In contrast, with the weighted POD approach the ROM predictions and FOM c_l ground truths are close to 0.4 for each reduced basis size. This highlights the potential of the weighted POD approach to produce more accurate flow field estimates, resulting in more accurate force coefficients.

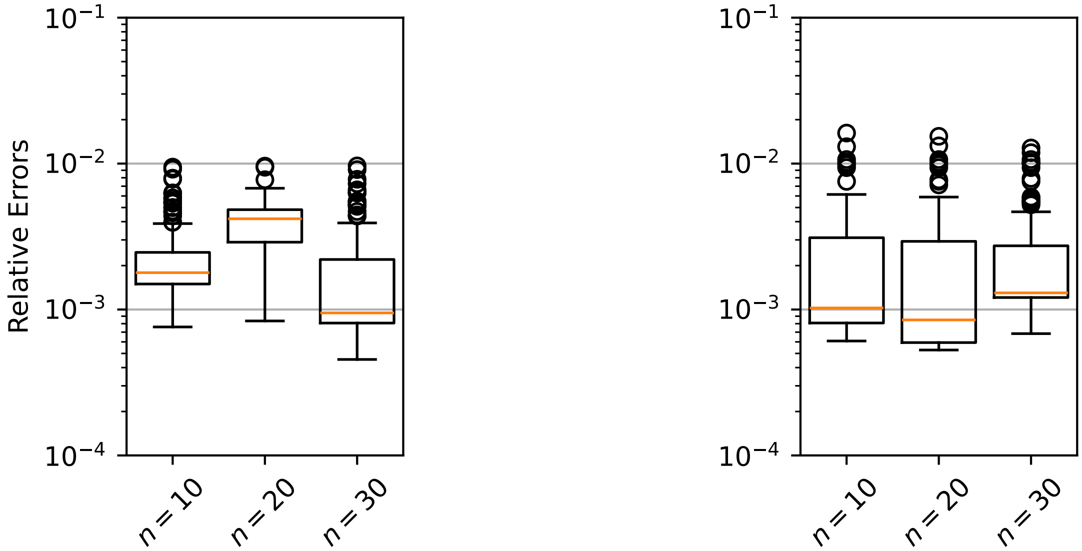
Figure 4 shows the relative L_2 errors of the global and weighted POD approaches; that is, the L_2 errors divided by the L_2 norm of the respective FOM solutions. As can be seen the median errors all vary closely around 10^{-5}. This seems to be the case because in most of the domain the POD solutions match the FOM solutions well. No clear trends are visible when increasing n. This is not very surprising, since Table 1 and Table 2 showed that both methods arrived at different optimal designs for different n. Figure 5 shows the relative L_2 errors when integrating over only the airfoil surface. We see that the relative errors grow to $O(10^{-3})$. Furthermore, the weighted POD approach with any reduced basis size shows comparable performance to the global POD basis with $n = 30$.

Next we look at the relative errors when predicting the lift and drag coefficients. Figure 6 shows the relative lift coefficient errors, i.e. the lift coefficient errors normalized by the FOM lift coefficients. As can be seen, both POD approaches produce similar relative lift coefficient errors. Again, no trend emerges when comparing the different values of n to one another, due to each optimization run taking a different path through the parameter space. When looking at the relative drag coefficient errors in Figure 7 a slightly different picture emerges. Since the drag coefficients are two orders of magnitude smaller than the lift coefficients, small local POD approximation errors on the airfoil surface can have a big impact on the approximated drag coefficients, especially near the leading edge. The weighted POD approach slightly outperforms the global POD approach for each considered reduced basis size.

Lastly we compare the wall times of both POD approaches. These are shown in Figure 8. As is to be expected, the wall time of the global POD basis goes up as n is increased, nearing the FOM simulation wall time, thereby becoming less useful as a way to speed up model evaluations. On the other hand, the weighted POD basis is consistently at least twice as fast as the FOM. This happens because of the approach with which the snapshot weights are assigned: If fewer than n snapshots are assigned a nonzero weight, the reduced basis that is used will be smaller than n.

VI. Conclusions

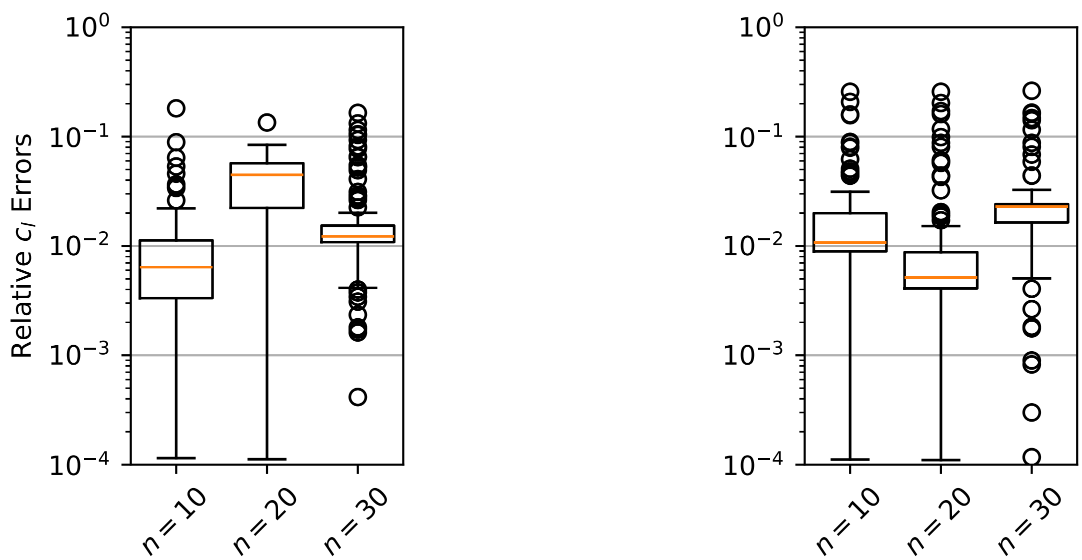
We present our first steps towards a novel hyper-reduced POD approach for large-scale aerodynamic design optimization. In this work we focus on applying a parametric Proper Orthogonal Decomposition (POD) approach to airfoil shape optimization problems, as well as laying the groundwork for the future steps of this research project by establishing the accuracy of a Discontinuous Galerkin (DG) model for compressible inviscid flow and implementing



(a) Global POD

(b) Weighted POD

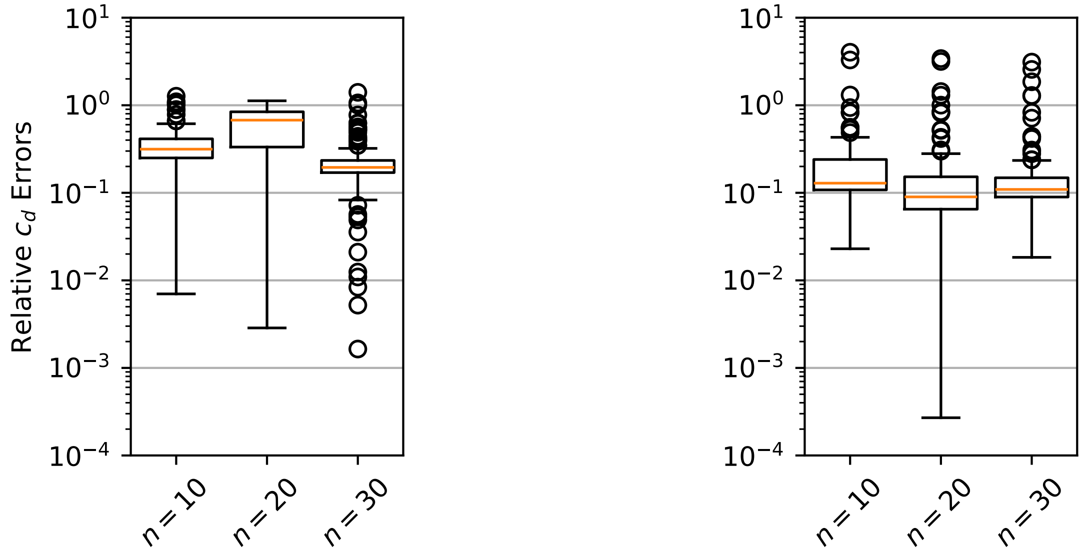
Fig. 5 Relative L_2 errors on the airfoil surface, obtained with the global and weighted POD approaches



(a) Global POD

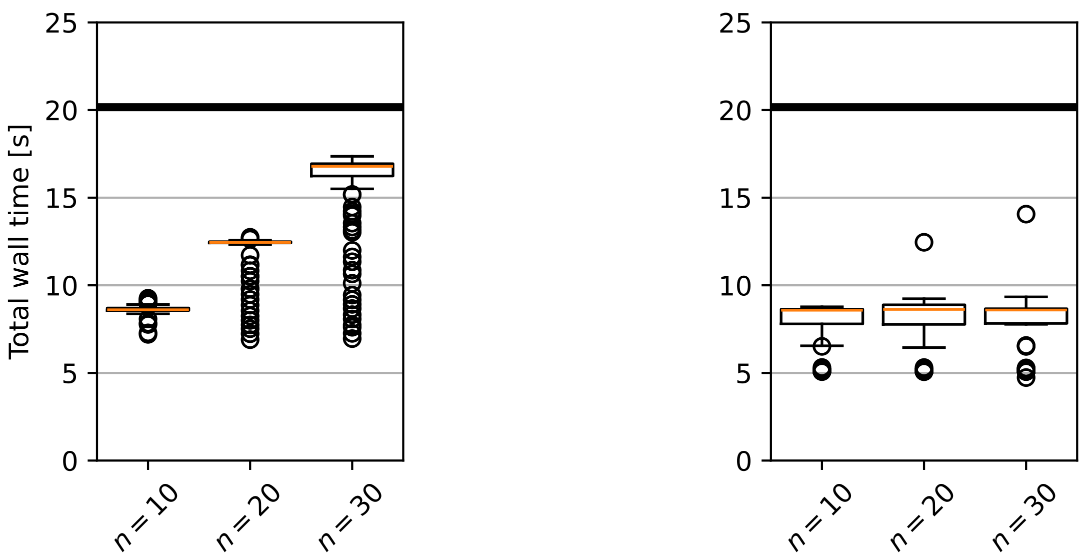
(b) Weighted POD

Fig. 6 Relative lift coefficient prediction errors of the global and weighted POD approaches



(a) Global POD **(b) Weighted POD**

Fig. 7 Relative drag coefficient prediction errors of the global and weighted POD approaches



(a) Global POD **(b) Weighted POD**

Fig. 8 Wall times of the global and weighted POD approaches. The black line indicates the average FOM wall time

a shape optimization pipeline. We show how the weighted POD approach that is proposed in [11] can be used for static nonlinear problems, and demonstrate how it especially improves the prediction of small coefficients such as drag coefficients. Moreover, we show how similar performance to the global POD basis is obtained in other metrics, but at a significantly shorter computational time.

While promising, the work presented here constitutes only the first few steps of a more elaborate research project into model reduction for large-scale design optimization. We intend to extend the current model reduction approach with a hyper-reduction step, in which the construction of Jacobian matrices for nonlinear solvers is approximated in order to speed up reduced models. Furthermore, in order to make it feasible to solve design optimization problems with hundreds or more design variables, we aim to use gradient-based optimization methods instead of the current gradient-free algorithms. Since our optimization pipeline is set up through the Computational Systems Design Language (CSDL) [18] and uses modOpt [19] as optimization package, switching to a different optimizer is straightforward and only requires the partial derivatives of the mesh morphing model and DG model to be made available. This would also allow us to leverage the second of the parametric POD methods that is proposed in [11], which uses gradient information to improve the accuracy of the POD basis.

Acknowledgments

The material presented in this paper is supported by NASA under award No. 80NSSC21M0070.

References

[1] Cliff, S. E., Reuther, J. J., Saunders, D. A., and Hicks, R. M., "Single-Point and Multipoint Aerodynamic Shape Optimization of High-Speed Civil Transport," *Journal of Aircraft*, Vol. 38, No. 6, 2001, pp. 997–1005. https://doi.org/10.2514/2.2886.

[2] Gray, A. C., and Martins, J. R., "Aerostructural Optimization of the Simple Transonic Wing Using MPhys, ADflow, and TACS," *AIAA SCITECH 2025 Forum*, 2025. https://doi.org/10.2514/6.2025-2813.

[3] Reuther, J. J., Jameson, A., Alonso, J. J., Rimlinger, M. J., and Saunders, D., "Constrained Multipoint Aerodynamic Shape Optimization Using an Adjoint Formulation and Parallel Computers, Part 1," *Journal of Aircraft*, Vol. 36, No. 1, 1999, pp. 51–60. https://doi.org/10.2514/2.2413.

[4] Reuther, J. J., Jameson, A., Alonso, J. J., Rimllnger, M. J., and Saunders, D., "Constrained Multipoint Aerodynamic Shape Optimization Using an Adjoint Formulation and Parallel Computers, Part 2," *Journal of Aircraft*, Vol. 36, No. 1, 1999, pp. 61–74. https://doi.org/10.2514/2.2414.

[5] Kenway, G. K. W., and Martins, J. R. R. A., "Multipoint High-Fidelity Aerostructural Optimization of a Transport Aircraft Configuration," *Journal of Aircraft*, Vol. 51, No. 1, 2014, pp. 144–160. https://doi.org/10.2514/1.C032150.

[6] Mader, C. A., and Martins, J. R. R. A., "Stability-Constrained Aerodynamic Shape Optimization of Flying Wings," *Journal of Aircraft*, Vol. 50, No. 5, 2013, pp. 1431–1449. https://doi.org/10.2514/1.C031956.

[7] Mendonça, G., Afonso, F., and Lau, F., "Model order reduction in aerodynamics: Review and applications," *Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering*, Vol. 233, No. 15, 2019, pp. 5816–5836. https://doi.org/10.1177/0954410019853472.

[8] Barone, M., Segalman, D., Thornquist, H., and Kalashnikova, I., "Galerkin Reduced Order Models for Compressible Flow with Structural Interaction," *AIAA SCITECH 2008 Forum*, 2008. https://doi.org/10.2514/6.2008-612.

[9] Rowley, C. W., Colonius, T., and Murray, R. M., "Model reduction for compressible flows using POD and Galerkin projection," *Physica D: Nonlinear Phenomena*, Vol. 189, No. 1, 2004, pp. 115–129. https://doi.org/10.1016/j.physd.2003.03.001.

[10] Parish, E. J., and Rizzi, F., "On the impact of dimensionally-consistent and physics-based inner products for POD-Galerkin and least-squares model reduction of compressible flows," *Journal of Computational Physics*, Vol. 491, 2023, p. 112387. https://doi.org/10.1016/j.jcp.2023.112387.

[11] Van Schie, S. P. C., Kramer, B., and Hwang, J. T., "Weighted Proper Orthogonal Decomposition for High-Dimensional Optimization," , 2025. Manuscript in preparation.

[12] Houston, P., and Sime, N., "Automatic Symbolic Computation for Discontinuous Galerkin Finite Element Methods," *SIAM Journal on Scientific Computing*, Vol. 40, No. 3, 2018, pp. C327–C357. https://doi.org/10.1137/17M1129751.

[13] Baratta, I. A., Dean, J. P., Dokken, J. S., Habera, M., Hale, J. S., Richardson, C. N., Rognes, M. E., Scroggs, M. W., Sime, N., and Wells, G. N., "DOLFINx: the next generation FEniCS problem solving environment," preprint, 2023. https://doi.org/10.5281/zenodo.10447666.

[14] Hesthaven, J. S., and Warburton, T., *Nodal Discontinuous Galerkin Methods: Algorithms, Analysis and Applications*, No. 54 in Texts in Applied Mathematics, Springer, 2008.

[15] Hartmann, R., and Houston, P., "Adaptive Discontinuous Galerkin Finite Element Methods for the Compressible Euler Equations," *Journal of Computational Physics*, Vol. 183, No. 2, 2002, pp. 508–532. https://doi.org/10.1006/jcph.2002.7206.

[16] Feng, L., Chellappa, S., and Benner, P., "A posteriori error estimation for model order reduction of parametric systems," *Advanced Modeling and Simulation in Engineering Sciences*, Vol. 11, No. 5, 2024. https://doi.org/10.1186/s40323-024-00260-8.

[17] Zahr, M. J., and Farhat, C., "Progressive construction of a parametric reduced-order model for PDE-constrained optimization," *International Journal for Numerical Methods in Engineering*, Vol. 102, No. 5, 2015, pp. 1111–1135. https://doi.org/10.1002/nme.4770.

[18] Gandarillas, V., Joshy, A. J., Sperry, M. Z., Ivanov, A. K., and Hwang, J. T., "A graph-based methodology for constructing computational models that automates adjoint-based sensitivity analysis," *Structural and Multidisciplinary Optimization*, Vol. 67, No. 76, 2014. https://doi.org/10.1007/s00158-024-03792-0.

[19] Joshy, A. J., and Hwang, J. T., "modOpt: A modular development environment and library for optimization algorithms," *Advances in Engineering Software*, Vol. 213, 2026, p. 104084. https://doi.org/10.1016/j.advengsoft.2025.104084.

[20] Harris, C. D., "Two-dimensional aerodynamic characteristics of the NACA0012 airfoil in the Langley 8 foot transonic pressure tunnel," Technical Memorandum NASA-TM-81927, NASA Langley Research Center, 1981.