This is a preprint of the following article, which is available from mdolab.engin.umich.edu
Neil Wu, Charles Mader and Joaquim R. Martins. "Sensitivity-based Geometric Parameterization for Aerodynamic Shape Optimization", AIAA 2022-3931. *AIAA AVIATION 2022 Forum*. June 2022.
The original article may differ from this preprint and is available at
https://arc.aiaa.org/doi/10.2514/6.2022-3931.

Sensitivity-based Geometric Parameterization for Aerodynamic Shape Optimization

Neil Wu, Charles A. Mader, and Joaquim R. R. A. Martins
University of Michigan, Ann Arbor, Michigan, 48109

Abstract

Aerodynamic shape optimization has become a well-established process, with designers routinely performing wing and full aircraft optimizations with hundreds of geometric design variables. However, with increased geometric design freedom comes increased optimization difficulty. These optimizations tend to converge very slowly, often taking many hundreds of design iterations. In addition, designers have to manually obtain suitable design variable scaling through trial and error in order to have a well-behaved optimization problem, which is a tedious and time-consuming task. In this work, we propose a sensitivity-based geometric parameterization approach that, while keeping the same optimization problem, maps the design space onto one which is better suited for gradient-based optimization. At the same time, we can automatically determine appropriate design variable scaling such that the new optimization problem can be solved more rapidly. We demonstrate the approach on aerodynamic optimizations, and show improved convergence behaviour compared to the traditional approach.

1 Introduction

There has been considerable advancement in aerodynamic shape optimization of wings and aircraft, accelerated by improvements in the analysis and optimization algorithms. It is now relatively routine to perform optimizations based on Reynolds-averaged Navier–Stokes (RANS) equations involving hundreds of geometric design variables, subject to multiple flight conditions and a large number of geometric constraints.

To this end, several geometric parameterizations have been developed and used in aerodynamic shape optimizations. While approaches such as Hicks-Henne bump functions and Class-Shape function Transformation (CST) are common for two-dimensional problems [1], for three-dimensional problems the most popular approach is to use free-form deformations (FFD) [2]. Instead of parameterizing the geometry directly, FFD parameterizes the geometric *deformation*, which offers some significant benefits [3]. Several popular aerodynamic shape optimization suites such as SU2[4], Jetstream [5, 6], and MACH[7] all use FFD for geometric parameterization.

Within MACH, we start by embedding a cloud of points called a "point set", which typically includes the surface mesh coordinates for aerodynamics or the finite element mesh nodes for structures. These points are then manipulated via geometric operations on the FFD nodes, which then apply the deformation to the embedded points. Instead of allowing each individual FFD node

to move arbitrarily in 3-dimensional space, we instead group them into intuitive sets of design variables. These operations can grow to be rather complicated, and can include nested geometric operations. Some examples include twist variables, which are rotations of a slice of FFD nodes at a spanwise section around a pre-defined rotation axis, and shape variables which are vertical displacements of individual FFD nodes. This approach allows for both global and local shape control, and the design variables represent quantities which are familiar to designers.

However, optimization performances are degraded significantly with the inclusion of complex geometric variables. For example, Lyu *et al.* performed an aerodynamic shape optimization on the Common Research Model (CRM) with 720 shape variables [8], and the initial optimization on a coarser grid took over 600 iterations. Even then, the best optimality achieved was only around 10^{-4}, and the overall optimization progress was slow. Similar trends have also been observed in other works, such as those by Koo and Zingg [9], Streuber and Zingg [10], and Kedward *et al.* [11]. This suggests that the scaling of the design variables could be improved, or that the design variables themselves cause poor conditioning of the optimization problem.

To address this issue, a class of adaptive geometric parameterizations has emerged. Bons *et al.* [12] employed a manual approach, where a sequence of optimizations is performed using successively-refined FFD control volumes which are defined *a priori*. This way, a significant amount of geometric changes can be achieved with the smaller design space, before switching to the denser FFD volume for finer geometric adjustments. Streuber *et al.* [10] on the other hand investigated adaptive geometric control, where the refinement of geometric parameterization is determined on-the-fly through the solution of an optimization sub-problem. By adaptively choosing the location of the geometric design variables during refinement, both computational speedup and superior optima were observed compared to the static approach.

However, another challenge that remains unsolved is in determining appropriate optimization scaling factors for geometric design variables. While design variable scaling is a general problem in optimization, geometric design variables are of particular interest because they comprise most of the design variables in aerodynamic shape optimization. Furthermore, in multidisciplinary design optimization (MDO) such as aerostructural applications, geometric variables affect both disciplines at the same time and appear as a dense sub-block in the constraint Jacobian. Historically, design variable scaling has been hand-tuned through trial-and-error, and the success and performance of an optimization can be largely dependent on this step. Given that typical geometric parameterizations consist of groups of disparate variables responsible for both local and global shape control, it is usually a significant challenge to find suitable scaling for them.

The aim of this work is to develop an approach which addresses numerical issues caused by these geometric design variables, but from an optimization perspective. We do not attempt to adaptively move or add geometric design variables, but work with the parameterization given. In this way, we do not alter the optimization problem or the design space of interest. Instead, we compute design variable mapping and scaling such that the optimization problem—while mathematically identical—becomes easier to solve. In the following sections, we outline the motivation behind our approach and the methodology itself. We then demonstrate the approach on an aerodynamic shape optimization problem, and show that the modified optimization problem results in improved optimizer performance.

2 Motivation

2.1 Impact of geometric design variables

To highlight the issue with geometric design variables further, we perform two aerodynamic shape optimizations where we minimize the total drag of a wing under transonic flight conditions. We perform the two RANS-based optimizations starting from the same baseline design; they differ only in the number of geometric design variables and constraints considered in the optimization problem. In one case, we consider only sectional twist variables, while in the other, additional shape variables are added. These shape variables are allowed to individually move in the vertical direction, and are typically used to improve the sectional airfoil shape. The geometry and FFD box used are shown in fig. 1. The flight condition is at Mach 0.8 and an altitude of 10 000 m. We use SNOPT [13] as the optimizer, and set both the feasibility and optimality tolerances to 10^{-6}. By converging the analyses and adjoing solutions tightly, we ensure that the optimization is fully converged and that those tolerances are achieved. The optimization problem is listed in table 1, where `T` refers to the twist-only optimization, and `T+S` the case including shape variables.

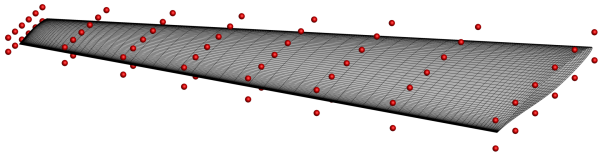


Figure 1: The standalone wing considered, together with the FFD box.

Table 1: The two aerodynamic shape optimization problems studied. "(L)" denotes that the constraint is linear.

	Function/variable	Description	Quantity T	Quantity T+S
minimize	C_D	Coefficient of drag		
with respect to	x_{twist}	Sectional twist	7	7
	x_{alpha}	Angle of attack	1	1
	x_{shape}	Shape variables	—	96
		Total design variables	**8**	**104**
subject to	$C_L = 0.5$	Lift constraint	1	1
	$V \geq V_{\text{base}}$	Minimum volume constraint	—	1
	$t \geq t_{\text{base}}$	Minimum thickness constraint	—	100
	$\Delta z_{\text{LE, upper}} = -\Delta z_{\text{LE, lower}}$	Fixed leading edge constraints (L)	—	8
	$\Delta z_{\text{TE, upper}} = -\Delta z_{\text{TE, lower}}$	Fixed trailing edge constraints (L)	—	8
		Total constraints	**1**	**118**

The optimization performance is shown in fig. 2. As expected, the optimization with only twist converged very quickly. In particular, once the approximate Hessian became sufficiently accurate, we observe the rapid convergence trends expected of quasi-Newton optimizations, reminiscent of Newton's methods. The last four iterations decreased the optimality tolerance by about an order of magnitude per iteration, dropping from 3×10^{-4} to 4×10^{-7} in that time.

On the other hand, the optimization with twist and shape took significantly longer, mirroring of the results from Lyu *et al.* [8]. Although it's expected that an optimization with more design variables will take more iterations, it took over 300 iterations in total to reach the same target

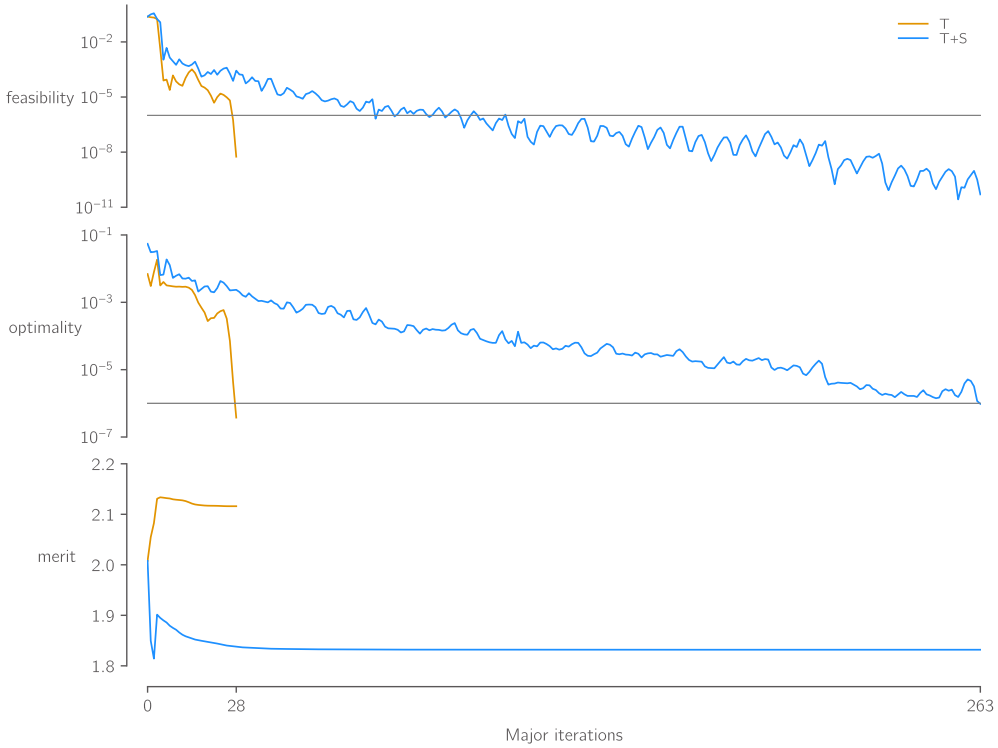


Figure 2: Comparison of aerodynamic shape optimizations with and without shape variables, showing various optimization metrics. The horizontal lines show the optimization tolerances, set at 10^{-6} for both.

optimality. From roughly iteration 100 onwards, progress became painfully slow, with very little improvement in either optimality or the merit function.

2.2 Impact of orthogonality

Given the stark contrast in performance between the two optimizations, we suspect that part of the issue lies in the geometric parameterization, where certain geometric design variables are not *orthogonal* to other design variables. This means that some subset of geometric design variables, while linearly independent, are in fact quite similar. Changing them would affect the outputs, in this case the embedded aerodynamic surface mesh nodes, in similar ways. In turn, this would have an adverse effect on the optimization as it causes poor conditioning for the optimization problem.

The earlier work by Lyu *et al.* [8] had examined the effect of changing the number of FFD nodes, both in the spanwise and chordwise directions. They had found that the addition of nodes in the spanwise direction had little effect on the number of optimization iterations taken, but there is a significant impact in the chordwise direction. Based on this, the authors had concluded that "the coupled effects between design variables are much stronger between variables within an airfoil than between variables in different airfoils".

To investigate this further, we analyze the `T+S` optimization problem performed earlier. We first compute the geometric Jacobian of surface sensitivities

$$\mathbf{J}_{\text{geo}} \triangleq \frac{\mathrm{d}\boldsymbol{X}_s}{\mathrm{d}\boldsymbol{x}_{\text{geo}}}, \tag{1}$$

where $\boldsymbol{X}_s$ is the flattened 1-D vector of surface mesh coordinates corresponding to the surface mesh of the wing used in CFD, and $\boldsymbol{x}_{\text{geo}}$ are the geometric design variables. Each column of this Jacobian

matrix represents the sensitivity of all surface mesh coordinates with respect to one geometric design variable. We examine the geometric sensitivity of surface mesh coordinates because they are the points embedded in the FFD. The volume mesh coordinates are then deformed by propagating these surface deformations using an inverse-distance approach, which is described in section 4.

From this, we can compute the angle between pairs of Jacobian columns $\boldsymbol{v}_i$ and $\boldsymbol{v}_j$ using eq. (2), and look for pairs of design variables which result in near parallel or antiparallel gradients, i.e. angles near 0° or 180°.

$$\theta_{ij} = \arccos\left(\frac{\boldsymbol{v}_i^T \boldsymbol{v}_j}{|\boldsymbol{v}_i| \cdot |\boldsymbol{v}_j|}\right) \tag{2}$$

where $\boldsymbol{v}_i$ is the i-th column of the Jacobian $\mathbf{J}$, and of course $i \neq j$.

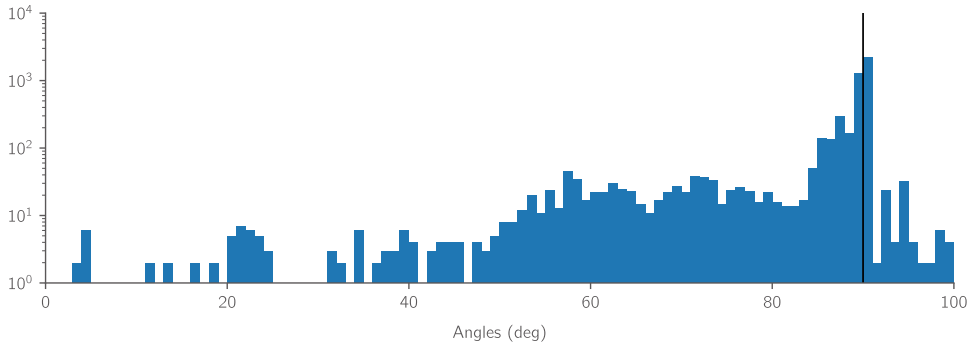


Figure 3: A histogram showing the distribution of angles between pairwise geometric design variable gradients. Note that the y-axis uses a logarithmic scale, and the vertical line is at 90°.

A histogram of the distribution of angles is given in fig. 3. We first notice that although the vast majority of design variable pairs are close to being orthogonal, there are some design variables which are very close to being "parallel" to each other, with the smallest angle at just 3.4°. A further check revealed that these correspond to two local shape variables that control the pair of upper and lower FFD nodes at the trailing edge of the wing tip, shown in fig. 4. In fact, in the top 18 pairs of non-orthogonal design variables we analyzed, they all correspond to upper and lower FFD nodes at the same planform location along the wing, with the ones most parallel at the trailing edge locations. This is expected, as these design variables are restricted to only move in the vertical direction, and moving the upper node would have a similar effect to moving the lower node, effectively displacing surface nodes near the FFD node in the vertical direction. These pairs of nodes are also more likely located at the trailing edge because at those locations, the surface mesh nodes are roughly equidistant to both the upper and lower FFD nodes. As a result, the surface sensitivities from either node are roughly equal.

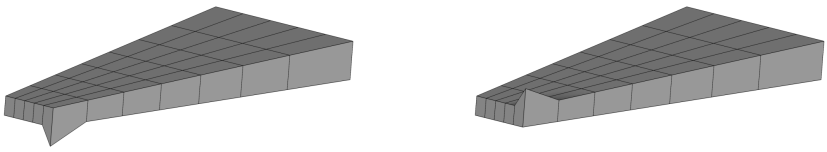


Figure 4: Shape variables 6 and 7, which have the smallest gradient angle at just 3.4°.

Now, it should be noted that in aerodynamic shape optimization, we employ a set of leading-edge and trailing-edge constraints which link precisely these pairs of design variables at the leading and trailing edges, in order to prevent those nodes from emulating sectional twist via shear deformations. These are listed in table 1, and effectively eliminate one of the two design variables in each pair. Unfortunately, some pairs of design variables in the interior of the wing, i.e. not at the leading or trailing edges, are still rather non-orthogonal. Out of these, the most non-orthogonal pair is shown in fig. 5. Lastly, there are also non-orthogonalities between certain shape and twist design variables, an example is shown in fig. 6.

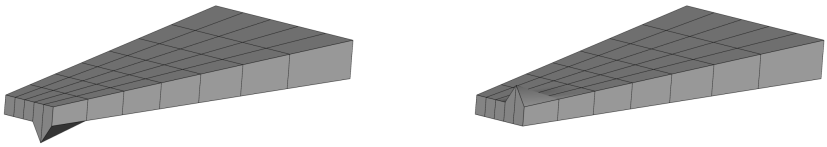


Figure 5: Shape variables 19 and 23, which have a gradient angle of 12°.

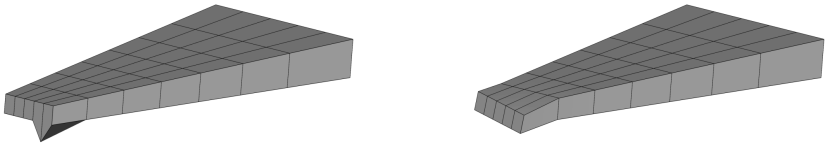


Figure 6: Shape variable 19 and twist variable 6, which have a gradient angle of 22°.

We then cross-check these potentially-problematic pairs of design variables against their optimization progress in fig. 7. Since these variables are at the leading or trailing edge, the linear constraint ensures that the pairs have equal and opposite values as SNOPT satisfies all linear constraints at each iteration. However, these design variables have noticeably slower convergence than the rest, which typically arrive close to their final value well within the first 100 iterations. Instead, these are still changing after 200 iterations, and it's likely that these variables are partly responsible for the slow overall convergence rate. In fact, all the design variables which have gradual changes during the optimization are those with highly non-orthogonal gradients.

Although the linear constraints remove some degrees of freedom from the design space, these design variables still pose a challenge for the optimization. To illustrate, suppose that we have two variables, x_1 and x_2 which have similar gradients, i.e.

$$\frac{\mathrm{d}I}{\mathrm{d}x_1} \approx \frac{\mathrm{d}I}{\mathrm{d}x_2}$$

for some objective $I(x_1, x_2)$. Then, suppose we add a linear constraint $x_1 + x_2 = 0$, meaning the two variables must be opposite of each other. This creates two opposing effects which render the design space rather flat. If moving along x_1 decreases I, then the corresponding increase in x_2 necessitated by the linear constraint would increase I by a similar amount since the gradients are similar. As a result, the sensitivity within the feasible space is very low, and the optimizer will have trouble finding the optimum. Fundamentally, the issue is caused by the fact that these design variables are not orthogonal to each other, resulting in similar gradients that worsen the conditioning of the optimization problem.

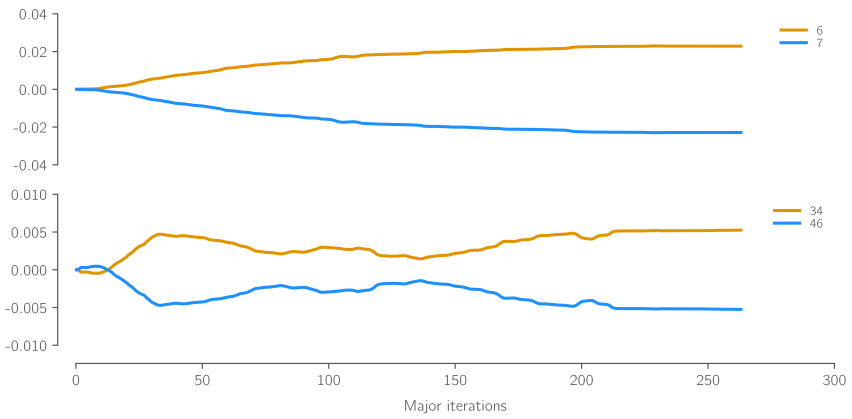


Figure 7: Optimization history for some shape variables at the leading and trailing edges.

Mathematically, we can show this by introducing a variable η which is used to parameterize the linear constraint:

$$x_1 = \eta, \quad x_2 = -\eta.$$

Then, the sensitivity along the constraint becomes

$$\begin{aligned} \frac{\mathrm{d}I}{\mathrm{d}\eta} &= \frac{\mathrm{d}I}{\mathrm{d}x_1}\frac{\mathrm{d}x_1}{\mathrm{d}\eta} + \frac{\mathrm{d}I}{\mathrm{d}x_2}\frac{\mathrm{d}x_2}{\mathrm{d}\eta} \\ &= \frac{\mathrm{d}I}{\mathrm{d}x_1} - \frac{\mathrm{d}I}{\mathrm{d}x_2} \\ &\approx 0 \end{aligned}$$

Optimization histories for some design variables which are not linearly constrained is shown in fig. 8. Even though no linear constraints exist, the fact that the gradients are similar means that the optimizer moves them in either equal or opposite directions, resulting in slow convergence.

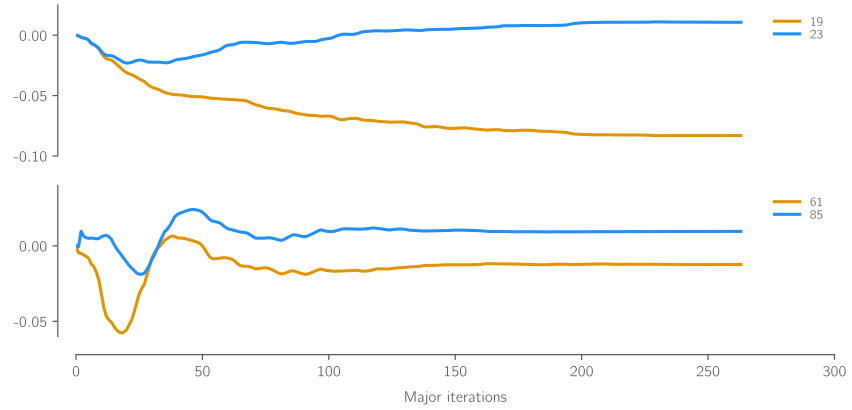


Figure 8: Optimization history for some shape variables at the interior of the wing.

3 Methodology

3.1 Generating design variables

We aim to map the existing geometric design variables $\boldsymbol{x}_{\text{geo}}$ into alternative variables which are more suitable for optimization. We do this through a linear mapping of the form

$$\hat{\boldsymbol{x}}_{\text{geo}} = \mathbf{A}\boldsymbol{x}_{\text{geo}}, \tag{3}$$

where $\boldsymbol{x}$ are the original design variables, and $\hat{\boldsymbol{x}}$ the new design variables. For the remainder of this sub-section, we omit the subscript "geo" with the implicit understanding that we are only performing this linear mapping on geometric design variables. The full optimization problem will likely have other design variables which remain unchanged. This is expanded further in 3.2

This matrix $\mathbf{A}$ is chosen such that the surface sensitivity Jacobian $\hat{\mathbf{J}}$ in the new design space, computed as

$$\hat{\mathbf{J}} \triangleq \frac{\mathrm{d}\boldsymbol{X}_s}{\mathrm{d}\hat{\boldsymbol{x}}} \tag{4}$$

has orthogonal columns. This means that, for two distinct design variables x_i and x_j, the dot product of the gradients should be zero:

$$\frac{\mathrm{d}\boldsymbol{X}_s}{\mathrm{d}\hat{\boldsymbol{x}}_i}^T \frac{\mathrm{d}\boldsymbol{X}_s}{\mathrm{d}\hat{\boldsymbol{x}}_j} = 0 \quad \forall i \neq j.$$

To do this, we first compute the geometric Jacobian $\mathbf{J}$, then perform singular value decomposition (SVD). This will rewrite the Jacobian as a multiplication of three matrices

$$\mathbf{J} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^T,$$

where $\mathbf{U}$ and $\mathbf{V}$ are comprised of the singular vectors, and $\boldsymbol{\Sigma}$ is a diagonal matrix of the singular values. By construction, $\mathbf{V}$ is an orthonormal matrix, and we use its inverse as the mapping matrix:

$$\mathbf{A} = \mathbf{V}^T. \tag{5}$$

The proposed methodology is a purely geometric approach based on the geometric sensitivities of embedded points, and does not fully capture the nonlinearity of the optimization problem. For example, the derivative of the objective with respect to the geometric design variables is given as

$$\frac{\mathrm{d}C_D}{\mathrm{d}\boldsymbol{x}} = \frac{\mathrm{d}C_D}{\mathrm{d}\boldsymbol{X}_s}\frac{\mathrm{d}\boldsymbol{X}_s}{\mathrm{d}\boldsymbol{x}},$$

and simply orthogonalizing the second term will not capture the effect of the CFD analysis. However, it should still offer some benefits for the optimization problem, especially since geometric design variables are often numerous and affect many optimization quantities at once. This is true both for multipoint problems where the same geometry is analyzed at different flight conditions, and for multidisciplinary problems where the disciplines share the same geometry. In fact, the proposed approach can be easily extended to aerostructural problems by including the embedded structural points in $\boldsymbol{X}_s$ when computing the Jacobian $\mathbf{J}$.

While there are similarities with modal approaches that also use SVD [14–16], there are some key differences. In the modal approach, SVD is applied on a set of training samples—typically candidate solutions from a library of candidate designs. For airfoil applications, the UIUC airfoil database is often used. In addition, dimension reduction is often applied to reduce the design space to only a handful of design variables. While effective for a range of applications, the modal approach

requires some *a priori* knowledge of the design space through the training samples, and operates on a modal design space that is a subspace of the full geometric design space permitted by the FFD approach. As a result, the optima found are usually inferior to the full-space optima, but at a reduced cost. On the other hand, the proposed approach does not alter the design space, and it is possible to obtain the same numerical optimum.

3.2 Reformulating the optimization problem

Once the mapping matrix $\mathbf{A}$ has been computed, the optimization problem has to be re-written in these new design variables $\hat{\boldsymbol{x}}_{\text{geo}}$. This involves mapping four aspects of the optimization problem:

- design variables
- derivatives of objectives and nonlinear constraints
- linear constraints
- design variable bounds

Additionally, since the mapping is done on the geometric design variables $\boldsymbol{x}_{\text{geo}}$ only, care must be taken to use the correct indices such that the rest of the design variables remain intact.

The first three items are straightforward to implement, where the design variables and the Jacobians need to be multiplied by $\mathbf{A}$ or its inverse. The design variable bounds, on the other hand, require some modification to the optimization problem itself. The bounds in the original space $\boldsymbol{x}$ are usually given as

$$\boldsymbol{x}_L \leq \boldsymbol{x} \leq \boldsymbol{x}_U, \tag{6}$$

but in the new design space, these bound constraints become linear inequality constraints:

$$\boldsymbol{x}_{L,g} \leq \mathbf{A}^{-1}\hat{\boldsymbol{x}}_{\text{geo}} \leq \boldsymbol{x}_{U,g}. \tag{7}$$

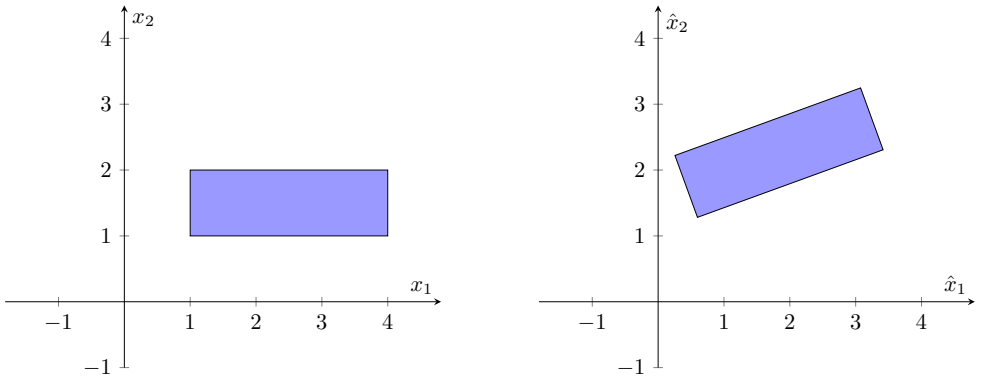
Since the matrix $\mathbf{A}$ is orthonormal, geometrically speaking, the design space mapping corresponds to a rotation and possibly a reflection. The original bound constraints form a rectangular feasible space, but after rotation, the boundary is no longer aligned with the coordinate axes of the new design space, as shown in fig. 9. Therefore, they must be replaced by explicit linear constraints, with $\mathbf{A}^{-1}$ as the linear Jacobian. Note that we only do this for the geometric design variables, and the rest of the design variable bounds are unchanged. While this technically increases the dimension of the optimization problem by introducing additional constraints, this should not cause any difficulties for gradient-based optimizers since the constraints are linear.

3.3 Verification

As before, we compute the angles θ_{ij} between pairs of columns of $\hat{\mathbf{J}}$ using eq. (2), but in the new design space $\hat{\boldsymbol{x}}$. Recall that in the original design space $\boldsymbol{x}$, the worst alignment has an angle of $\theta_{ij} = 3.4°$. Through the singular value decomposition, we expect all the gradient vectors to be orthogonal to each other. We then compute the maximum deviation from orthogonality, computed as

$$\Delta\theta_{\max} = \max_{\substack{i,j \\ i \neq j}} \left|\theta_{ij} - 90°\right| \tag{8}$$

As shown in table 2, the linear transformation does indeed yield an orthogonal Jacobian, with the maximum deviation close to machine precision.



(a) The original design space bounds.

(b) The same bounds in the new design space.

Figure 9: An illustration of the transformation of bound constraints under orthonormal mapping. The original feasible space is now rotated in the new axes, which can no longer be expressed as simple design variable bounds.

Table 2: Verification of the orthogonality of geometric sensitivities.

	$\Delta\theta_{\max}$ (deg)
$\boldsymbol{x}$	86.58
$\hat{\boldsymbol{x}}$	2.48×10^{-11}

3.4 Generated design variables

Finally, we show the first eight design variables in fig. 10, ranked by the largest singular values. We see that some of the design variables are recognizable, for example the 3th and 5th design variables show root camber deformations. However, as we move through, later design variables are less discernable, but that is expected in any decompositional approach. Since the transformation is linear, we can always map the design variables back and forth with relative ease, and users can monitor optimization progress using more physically-intuitive design variables.

3.5 Design Variable Scaling

One disadvantage of performing design variable mapping is that these constructed design variables have no physical meaning behind them. As a result, it's difficult to determine an appropriate scaling factor for each variable based on physical quantities. In addition, since these design variables have no bounds applied, the common technique of scaling them to be in the range of $[0, 1]$ cannot be used either.

However, there were also difficulties with the traditional approach to scaling. Physical intuitions can be used to deduce, for example, that the scaling for the angles of attack should be the same as for the spanwise twist variables on the wing, since they have the same units and roughly the same impact on the flow field. But it is unclear how to select the scaling factor across disparate design groups, such as shape or sweep. Users have to largely rely on trial and error to produce well-behaved optimizations, which can consume a significant amount of time. As an example, Bons [17] performed several large-scale aerostructural optimizations, where the scaling factors on various geometric design variables were hand-tuned.

Now, with the additional insight provided by the singular value decomposition, we can use that information to automatically determine the appropriate scaling for these design variables. In particular, the singular values σ_i corresponding to each design variable x_i are a natural choice. For

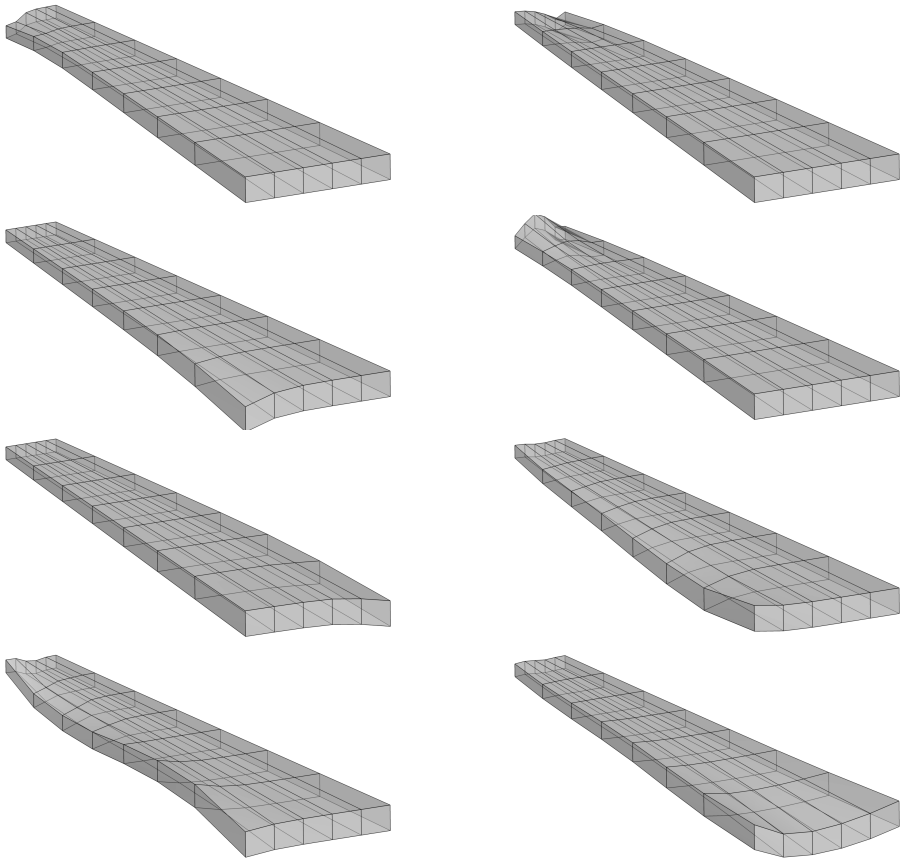


Figure 10: The first eight newly constructed design variables.

a given unit of change, the design variables with larger singular values have a larger impact on the surface mesh coordinates, and are therefore more sensitive.

We seek a scaling function $s(\sigma)$ that gives the appropriate scaling used for each design variable. Logically, we would like to reduce the sensitivity of more impactful design variables, and vice versa. This would ensure that all design variables have similar sensitivities, and therefore converge equally quickly. From a numerical optimization perspective, this should also ensure a better conditioned Hessian with similar curvatures along different axes [18].

In this work, we try four different choices:

- $s_i = 1$
- $s_i = \sigma_i$
- $s_i = \sqrt{\sigma_i}$
- $s_i = 1/\sigma_i$

As a larger scaling value will *decrease* the sensitivity of a design variable, we would expect monotonically increasing choices for $s(\sigma)$, such as $s = \sigma$ or $s = \sqrt{\sigma}$, to perform well.

4 Computational framework

We use the MACH-Aero Framework[1] to perform gradient-based aerodynamic shape optimizations. The aerodynamic analysis is performed using ADflow [19], a finite-volume CFD solver with an efficient adjoint implementation suitable for aerodynamic shape optimization. In this work, we use RANS with the Spalart–Allmaras (SA) turbulence model, and converge the solutions to a relative residual of 10^{-14}. The discrete adjoint equations are similarly converged by 10^{-14} relative to the initial linear residual. We converge these tightly to ensure that we are able to reach the target feasibility and optimality tolerances, without the optimizer wasting extra evaluations due to inaccurate function values or gradients.

We use pyGeo [7] to parameterize the geometry and deform the aerodynamic surface mesh using free-form deformation (FFD) volumes. We then use IDWarp [20] to warp the aerodynamic volume mesh based on the surface mesh deformations.

For optimization, we use pyOptSparse [21], a framework tailored for large-scale gradient-based optimizations. It provides wrappers for several popular non-linear optimization algorithms, and for this work we use SNOPT [13]. For all results presented in this paper, we set the feasibility and optimality tolerances to 10^{-6}. The feasibility refers to the satisfaction of the zeroth-order KKT conditions, and optimality the first-order conditions. Together, they give a measure of whether the design is close to the numerical optimum.

5 Results

We perform two different optimization problems: the `T+S` problem presented above, and one with an additional span variable which we call `T+S+S`. The optimizations involving span have an extra constraint such that the projected planform area of the wing must be greater than or equal to the initial planform area.

For each of these optimization problems, we perform five optimizations in order to compare the two parameterizations, and explore the different scaling options. We first perform a reference optimization using the existing parameterization, and four optimizations using the new parameterization with various design variable scalings. In order for this to be a fair comparison, we converge the CFD primal and adjoint solutions to the same level, and use the same optimization termination criteria. The only difference is in the design variable mapping and scaling that affects the performance of the optimizer. This way, we can check firstly if the four scaling options all converge to the same reference optimum, and subsequently analyze their performance. Table 3 lists the optimization problem for the `T+S` case, which shows that the new parameterization has the same number of design variables, and extra linear constraints are in place to replace the bound constraints in the original design space.

5.1 Twist and Shape

Table 4 shows the outcomes of those optimizations. As explained above, the mapping does not change the dimensionality of the problem, since the additional linear constraints are replacing the design variable bounds. Therefore, we would expect all the optimizations to arrive at the same optimum.

However, this was not the case. While most of the optimizations converged to the same optimum as the reference case without geometric mapping, the cases with $s = 1$ and $s = 1/\sigma$ did not. Although feasible, these two optimizations took significantly more iterations, and terminated with

[1] https://github.com/mdolab/MACH-Aero

Table 3: The optimization problem, highlighting the differences in problem size between the original and new parameterizations.

	Function/variable	Description	Quantity Original	New
minimize	C_D	Drag coefficient	1	1
with respect to	x_{twist}	Section twist	7	0
	x_{shape}	Shape	96	0
	x_{user}	User	0	103
	x_{alpha}	Angle of attack	1	1
		Total design variables	**104**	**104**
subject to	$t \geq t_0$	Thickness constraint	100	100
	$V \geq V_0$	Volume constraint	1	1
	$\Delta z_{\text{LE,upper}} = -\Delta z_{\text{LE,lower}}$	Fixed LE constraint (L)	8	8
	$\Delta z_{\text{TE,upper}} = -\Delta z_{\text{TE,lower}}$	Fixed TE constraint (L)	8	8
	$x_L \leq x \leq x_U$	Design variable bound (L)	0	103
	$C_L = 0.5$	Lift constraint	1	1
		Total constraints	**118**	**221**

worse objective values. In the case of $s = 1$, the optimization took the maximum allowable major iterations of 1000, and terminated before reaching the final optimality tolerance of 10^{-6}. This is somewhat consistent with our expectations, as those two scaling options did not reduce the sensitivity of more impactful design variables. This highlights the importance of design variable scaling which can have significant impact on the performance of the optimization.

Table 4: Summary of optimization results for the T+S problem.

Scaling	Objective (counts)	Major itns	Feasibility	Optimality	Time (s)	Improvement (%)
None	183.188	263	5.0×10^{-11}	9.6×10^{-7}	9398	—
$s = 1$	183.231	1000	9.1×10^{-9}	6.4×10^{-4}	35 799	−281
$s = \sigma$	183.188	309	9.7×10^{-12}	4.7×10^{-7}	11 806	−26
$s = \sqrt{\sigma}$	183.188	284	1.5×10^{-11}	9.8×10^{-7}	10 193	−8
$s = 1/\sigma$	183.198	577	6.4×10^{-13}	8.3×10^{-7}	21 561	−129

Out of the three optimizations that did converge to the same optimum, the use of design variable mapping did not offer a clear advantage over the existing parameterization. The best-performing scaling option, $s = \sqrt{\sigma}$, took 21 more major iterations and 8% longer wall time to converge. However, it's worth pointing out that the reference optimization was performed with design variable scalings that were tuned through prior experience. Therefore, obtaining comparable performances is an achievement in and of itself, since it offers cost savings in the setup and tuning of the optimization that is not covered by the comparison above.

Next, we look at the history of the optimization in more detail. Figure 11 shows a few design variables over the course of the optimization. Although the shape and twist variables were not used directly, we can easily apply the inverse mapping to obtain their values throughout the optimization. From these plots, it's clear that the two optimizations with $s = 1$ and $s = 1/\sigma$ did not converge. Their design variable values stayed near the initial design for much of the optimization, and the final value does not match the reference optimum. For the remaining three optimizations, some variables converged more rapidly than the reference, while others exhibited more oscillations.

The optimization metrics are shown in fig. 12. In addition to the previously-mentioned metrics,

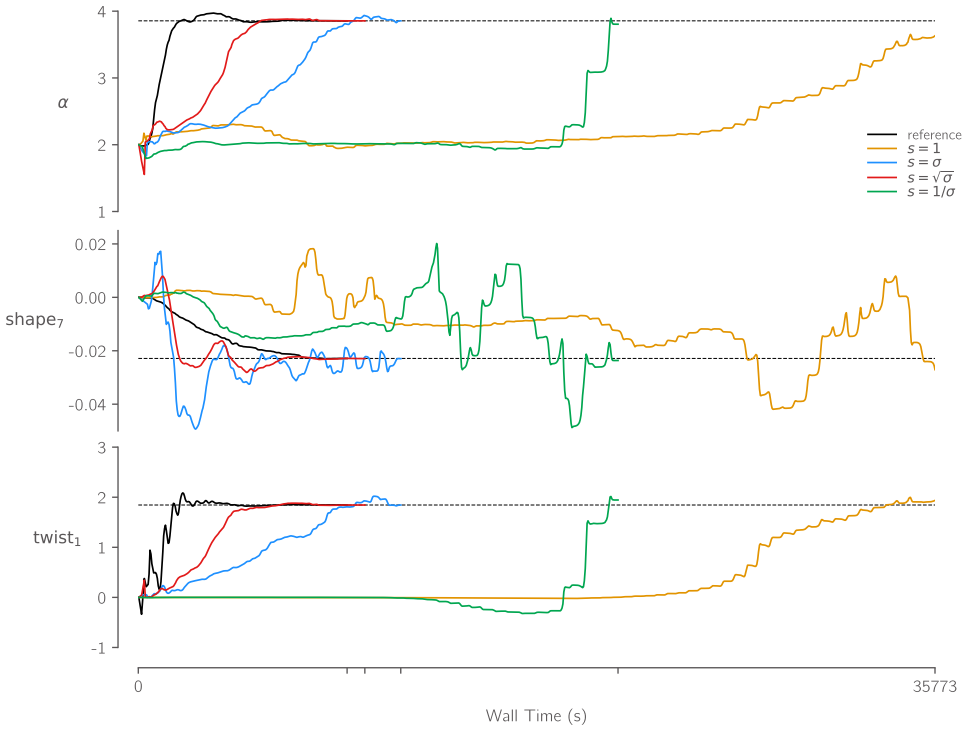


Figure 11: The design variable history for each optimization. The dashed line corresponds to the final value obtained from the reference optimization without any design variable mapping.

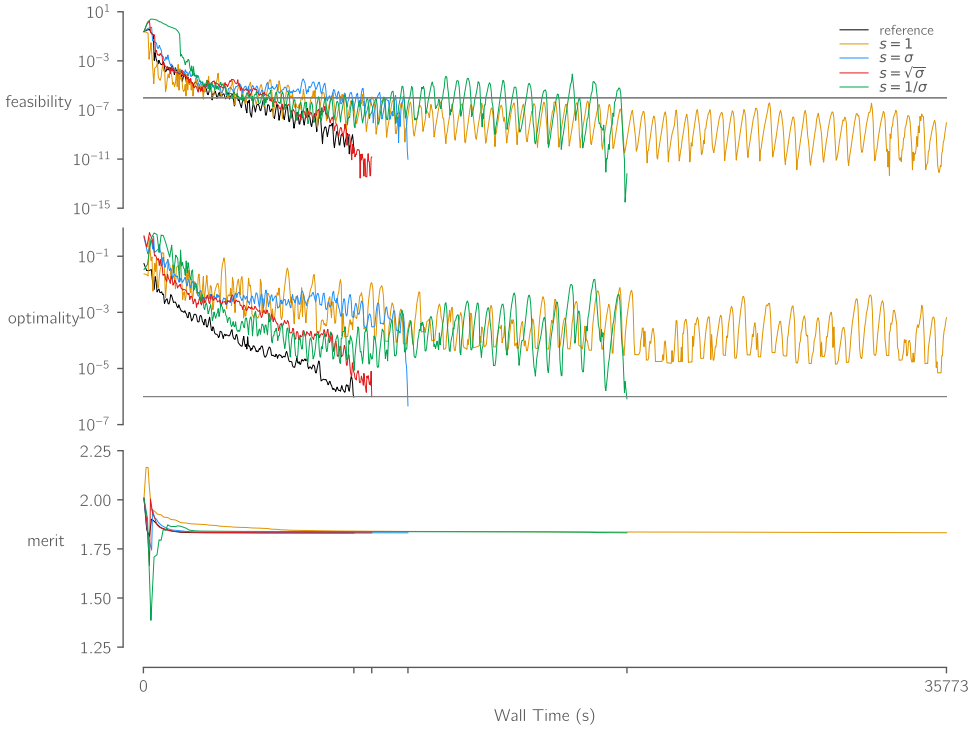


Figure 12: The optimization metrics for the T+S case. The thin horizontal black line indicates the termination criteria of 10^{-6} for feasibility and optimality.

we also show the merit function, which is an augmented Lagrangian composed of three terms: the objective, the constraints weighted by the Lagrange multipliers, and a quadratic penalty term on the constraint violation [13]. This metric is helpful in gauging optimization progress, as it combines the objective value with the constraints. Again, we can see that the two unconverged optimizations were much slower at reducing the merit function compared to the others. They also had large oscillations for feasibility and optimality, indicating difficulties in obtaining an accurate approximate Hessian.

On the other hand, the optimization with $s = \sigma$ was able to converge very rapidly at the end of its optimization, reminiscent of the twist-only case presented earlier. After close to 300 iterations with relatively little change to optimality, it was able to decrease it from 10^{-3} to 10^{-6} in 6 iterations. This rapid convergence rate is an indication that it was able to accumulate a sufficiently-accurate approximate Hessian to be within the quadratic convergence region, likely due to a combination of orthogonal design variables and suitable scaling. Despite taking longer than the reference optimization, the rapid terminal convergence means that for a tighter termination criterion, it could conceivably obtain an optimum in fewer iterations.

Lastly, we extract the approximate Hessian at the final iteration and show them in fig. 13. We ensure that the Hessian is never reset during the optimization, either periodically or when the optimizer encounters numerical difficulties, so that these plots are representative of the design space as perceived by the optimizer. In addition, the optimizations presented here all took a large number of major iterations, on the order of several times the number of design variables. Therefore, they represent a reasonable approximation to the Hessian of the Lagrangian, which gives us an idea of the curvature of the problem.

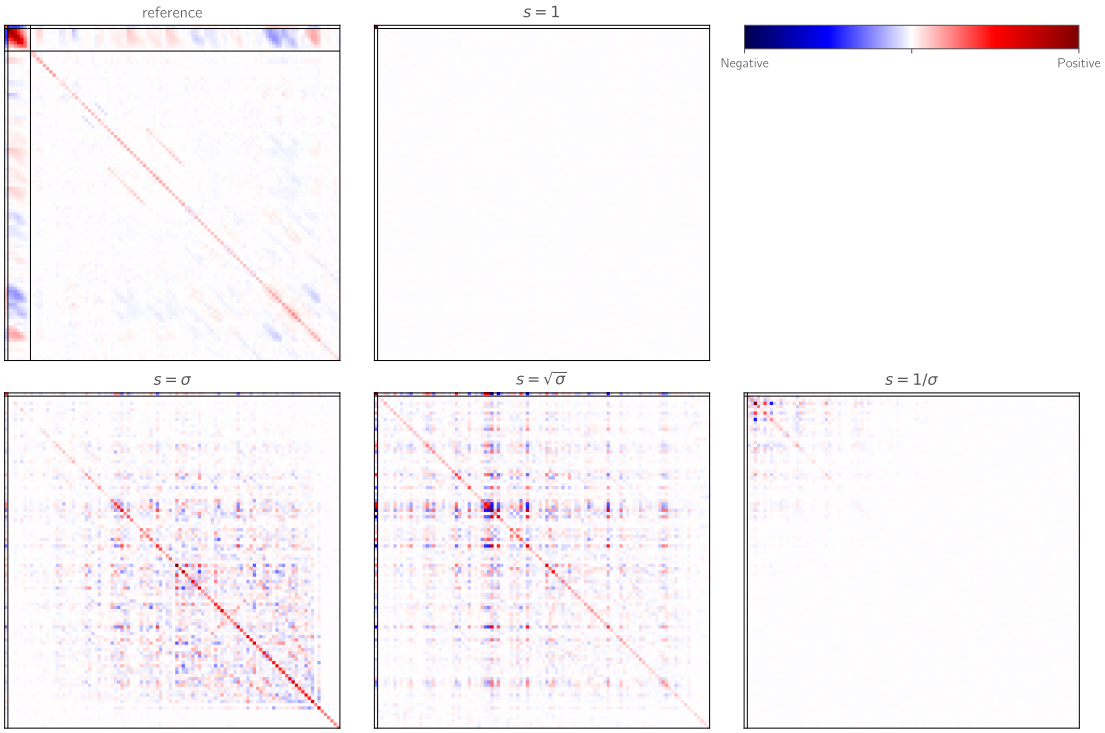


Figure 13: The approximate Hessian at the final optimization iteration. Note that while a symmetric diverging colormap is used, the maximum value is adjusted for each plot. Therefore, entries with the same color do not have the same magnitude across different subplots.

In these figures, the horizontal and vertical thin black lines are used to show different design

variable groups. For the reference optimization, the first row and column corresponds to the angle of attack variable, followed by the seven twist variables. The remaining entries are the shape variables. For the other optimizations, the design variable mapping groups all geometric design variable into a single sub-block, and the only other variable is α.

The structure of the approximate Hessian is clearly visible in the reference optimization, with the twist variables having the largest entries, particularly along the diagonal. Significant coupling across design variables can also be seen, in the structure of the off-diagonal terms. Many of these off-diagonal entries are precisely the design variable pairs that have near-parallel alignment, not just between the shape variables but with twist variables as well. When mapping is applied, we see that the structured pattern in the Hessian has been eliminated, and that different scaling options affect the magnitudes of the entries. For the two optimizations that did not converge, it is clear that the the approximate Hessian was far from accurate. For the remaining optimizations, the magnitudes of the diagonal terms are quite informative.

In the ideal scenario, a well-scaled optimization should have entries of similar magnitudes along the diagonal, which indicates similar curvatures across design variables. We see that with the scale option $s = \sigma$, there is an over-emphasis on the later design variables, whereas $s = \sqrt{\sigma}$ had a more uniform distribution. This could be the reason for this particular scaling to come closest to the reference optimization.

5.2 Span

Similar trends were observed for the optimization involving twist, shape, and span variables. The optimization results are summarized in table 5.

Table 5: Summary of optimization results for the T+S+S problem.

Scaling	Objective (counts)	Major itns	Feasibility	Optimality	Time (s)	Improvement (%)
None	171.924	373	8.0×10^{-12}	9.5×10^{-7}	13 506	—
$s = 1$	171.934	1000	2.4×10^{-12}	3.8×10^{-6}	37 219	−176
$s = \sigma$	171.924	353	2.7×10^{-15}	7.8×10^{-7}	14 612	−8
$s = \sqrt{\sigma}$	171.924	314	1.9×10^{-14}	8.6×10^{-7}	12 584	7
$s = 1/\sigma$	171.936	547	1.7×10^{-14}	7.0×10^{-7}	20 941	−55

The design variable histories are shown in fig. 14, and the optimization metrics are shown in fig. 15. As before, the scalings $s = 1$ and $s = 1/\sigma$ did not converge, and $s = \sqrt{\sigma}$ performed the best, beating the reference optimization by 7%. Interestingly, both $s = \sigma$ and $s = \sqrt{\sigma}$ took fewer major iterations to converge than the reference optimization, and the rapid terminal convergence stage was observed again in fig. 15.

The final approximate Hessians are shown in fig. 16, and once again we see that the $s = \sqrt{\sigma}$ case had a more uniform distribution of entries along the diagonal. In retrospect, this outcome makes a lot of sense. When we scale variables by a factor s:

$$\hat{x} = sx,$$

this causes the derivatives to be scaled by the same factor s. Since the Hessian is the matrix of second derivatives, this causes the entries of the Hessian to be scaled by s^2. If the aim is to scale those Hessian entries by σ, then scaling the design variables by its square root would be the logical choice.

The differences in performance between these four identical optimizations show the importance of design variable scaling. Together with the construction of the new design variables, we demonstrate

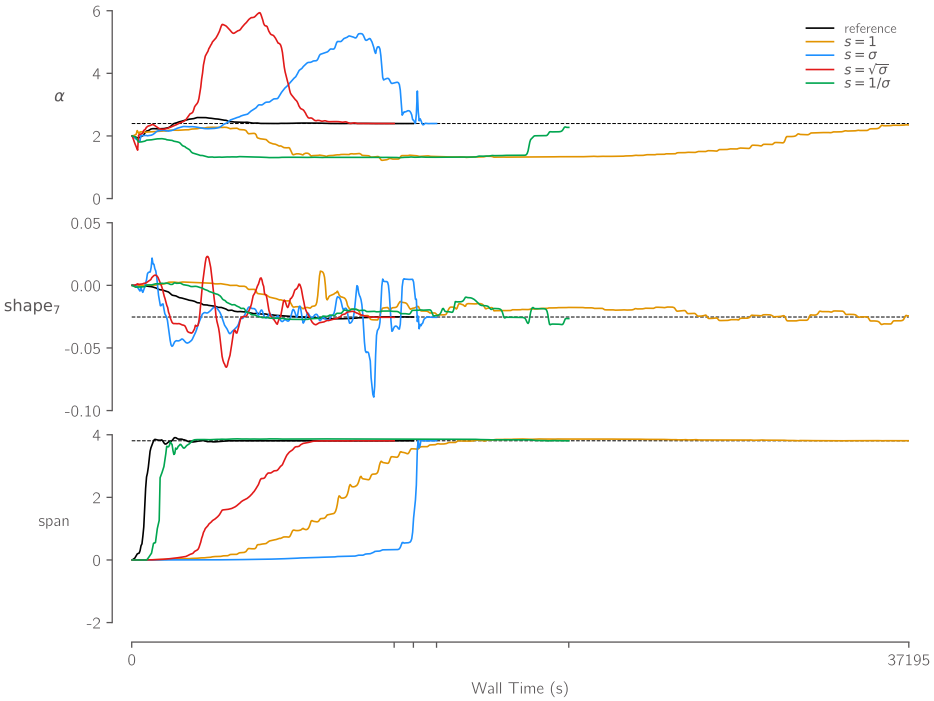


Figure 14: The design variable history for the T+S+S case.

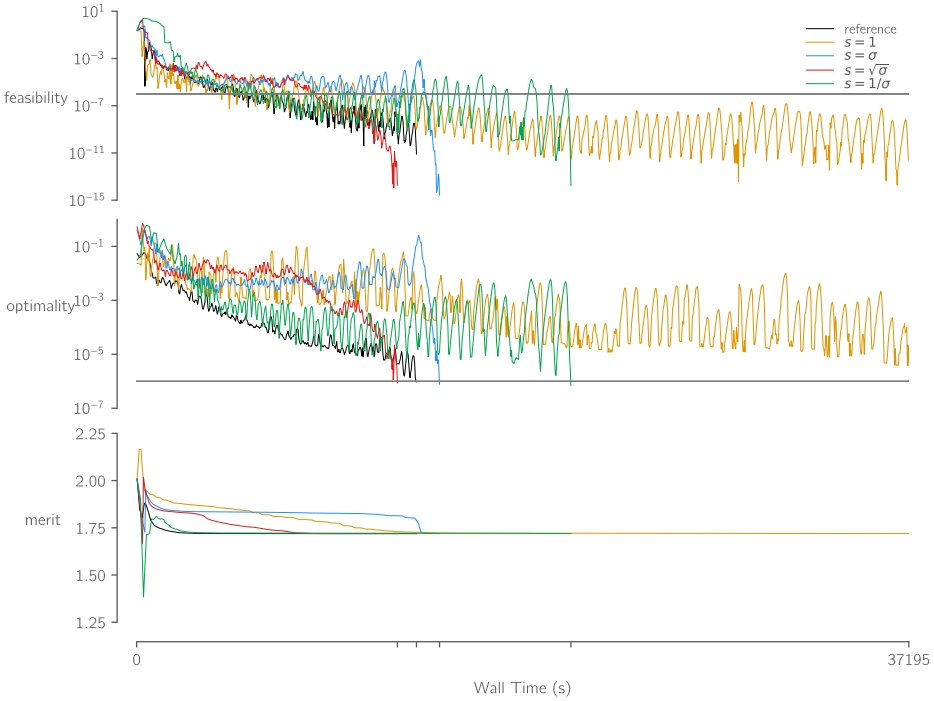


Figure 15: The optimization metrics for the T+S+S case.

that despite examining only the geometric properties of the optimization problem, we can improve the performance of the optimization.

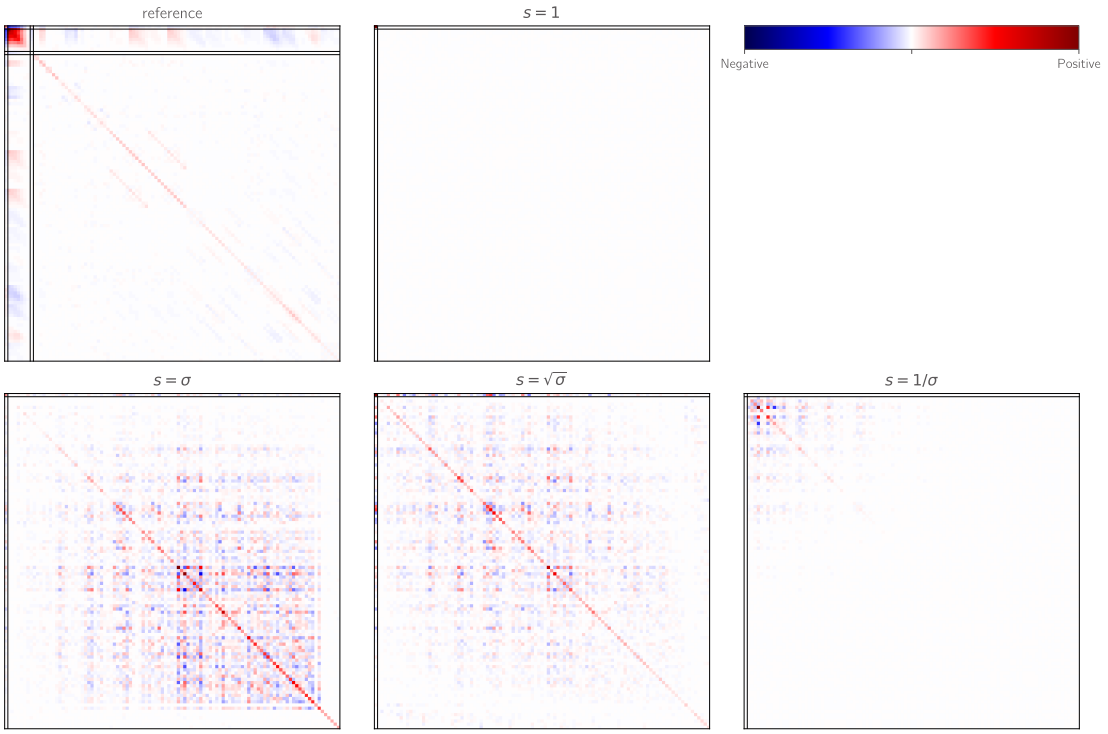


Figure 16: The approximate Hessian at the final iteration for the T+S+S case. The additional span variable is visible in the reference optimization at the 9th index, highlighted by the thin vertical and horizontal lines.

6 Conclusion

In this work, we showed that the existing geometric design variables based on free-form deformation are not orthogonal to each other. In some cases, the design variables affect the embedded surface mesh in very similar ways, which could lead to issues with the optimizer. To address this, we constructed a new set of design variables based on singular value decomposition of the geometric Jacobian, such that the Jacobian in the new design space is orthogonal. At the same time, we use the singular values from the SVD to automatically scale these constructed design variables.

We then perform two sets of aerodynamic shape optimizations, first with twist and shape variables, and later including span. For each optimization problem, we compared the original parameterization against the newly developed parameterization with various scaling approaches. We found that with the appropriate design variable scaling, the proposed methodology was able to improve the convergence behaviour of the optimization while converging to the same optimum. Unlike the reference optimization, we were able to eliminate the long optimization tail and recover the rapid terminal convergence expected of quasi-Newton methods. Although the overall computational costs were comparable to the reference optimization, the automatic scaling eliminates the need for manual tuning, which is a significant advantage over the traditional approach.

Acknowledgments

The material is based upon work supported by Airbus in the frame of the Airbus / Michigan Center for Aero-Servo-Elasticity of Very Flexible Aircraft. Special thanks to Tom Gibson and Joël Brezillon for their expert advice and review of this paper. This research was supported in part through computational resources and services provided by Advanced Research Computing at the

University of Michigan, Ann Arbor. This research also made extensive use of the Texas Advanced Computing Center (TACC) Stampede2 supercomputer via Extreme Science and Engineering Discovery Environment (XSEDE), which is supported by National Science Foundation grant number ACI-1548562.

References

[1] Castonguay, P., and Nadarajah, S. K., "Effect of Shape Parameterization on Aerodynamic Shape Optimization," *45th AIAA Aerospace Sciences Meeting and Exhibit*, Reno, Nevada, 2007. https://doi.org/10.2514/6.2007-59.

[2] Sederberg, T. W., and Parry, S. R., "Free-form Deformation of Solid Geometric Models," *SIGGRAPH Comput. Graph.*, Vol. 20, No. 4, 1986, pp. 151–160. https://doi.org/10.1145/15886.15903.

[3] Zhang, T.-t., Wang, Z.-g., Huang, W., and Yan, L., "A review of parametric approaches specific to aerodynamic design process," *Acta Astronautica*, Vol. 145, 2018, pp. 319–331. https://doi.org/10.1016/j.actaastro.2018.02.011.

[4] Economon, T. D., Palacios, F., Copeland, S. R., Lukaczyk, T. W., and Alonso, J. J., "SU2: An Open-Source Suite for Multiphysics Simulation and Design," *AIAA Journal*, Vol. 54, No. 3, 2016, pp. 828–846. https://doi.org/10.2514/1.j053813.

[5] Hicken, J. E., and Zingg, D. W., "Aerodynamic Optimization Algorithm with Integrated Geometry Parameterization and Mesh Movement," *AIAA Journal*, Vol. 48, No. 2, 2010, pp. 400–413. https://doi.org/10.2514/1.44033.

[6] Gagnon, H., and Zingg, D. W., "Two-Level Free-Form and Axial Deformation for Exploratory Aerodynamic Shape Optimization," *AIAA Journal*, Vol. 53, No. 7, 2015, pp. 2015–2026. https://doi.org/10.2514/1.J053575.

[7] Kenway, G. K., Kennedy, G. J., and Martins, J. R. R. A., "A CAD-Free Approach to High-Fidelity Aerostructural Optimization," *Proceedings of the 13th AIAA/ISSMO Multidisciplinary Analysis Optimization Conference*, Fort Worth, TX, 2010. https://doi.org/10.2514/6.2010-9231.

[8] Lyu, Z., Kenway, G. K. W., and Martins, J. R. R. A., "Aerodynamic Shape Optimization Investigations of the Common Research Model Wing Benchmark," *AIAA Journal*, Vol. 53, No. 4, 2015, pp. 968–985. https://doi.org/10.2514/1.J053318.

[9] Koo, D., and Zingg, D. W., "Investigation into Aerodynamic Shape Optimization of Planar and Nonplanar Wings," *AIAA Journal*, Vol. 56, 2018, pp. 250–263. https://doi.org/10.2514/1.j055978.

[10] Streuber, G. M., and Zingg, D. W., "Dynamic Geometry Control for Robust Aerodynamic Shape Optimization," *AIAA AVIATION 2021 Forum*, 2021. https://doi.org/10.2514/6.2021-3031.

[11] Kedward, L. J., Allen, C. B., and Rendall, T. C. S., "Gradient-Limiting Shape Control for Efficient Aerodynamic Optimization," *AIAA Journal*, Vol. 58, No. 9, 2020, pp. 3748–3764. https://doi.org/10.2514/1.j058977.

[12] Bons, N. P., and Martins, J. R. R. A., "Aerostructural Design Exploration of a Wing in Transonic Flow," *Aerospace*, Vol. 7, No. 8, 2020, p. 118. https://doi.org/10.3390/aerospace7080118.

[13] Gill, P. E., Murray, W., and Saunders, M. A., "SNOPT: An SQP Algorithm for Large-Scale Constrained Optimization," *SIAM Review*, Vol. 47, No. 1, 2005, pp. 99–131. https://doi.org/10.1137/S0036144504446096.

[14] Li, J., Bouhlel, M. A., and Martins, J. R. R. A., "Data-based Approach for Fast Airfoil Analysis and Optimization," *AIAA Journal*, Vol. 57, No. 2, 2019, pp. 581–596. https://doi.org/10.2514/1.J057129.

[15] Li, J., and Zhang, M., "Adjoint-Free Aerodynamic Shape Optimization of the Common Research Model Wing," *AIAA Journal*, Vol. 59, No. 6, 2021, pp. 1990–2000. https://doi.org/10.2514/1.j059921.

[16] Poole, D. J., Allen, C. B., and Rendall, T., "Efficient aeroelastic wing optimization through a compact aerofoil decomposition approach," *Structural and Multidisciplinary Optimization*, Vol. 65, No. 3, 2022, pp. 1–19. https://doi.org/10.1007/s00158-022-03174-4.

[17] Bons, N. P., "High-fidelity Wing Design Exploration with Gradient-based Optimization," Ph.D. thesis, University of Michigan, Ann Arbor, MI, May 2020.

[18] Gill, P. E., Murray, W., and Wright, M. H., *Practical Optimization*, Academic Press, 1981.

[19] Mader, C. A., Kenway, G. K. W., Yildirim, A., and Martins, J. R. R. A., "ADflow: An open-source computational fluid dynamics solver for aerodynamic and multidisciplinary optimization," *Journal of Aerospace Information Systems*, Vol. 17, No. 9, 2020, pp. 508–527. https://doi.org/10.2514/1.I010796.

[20] Secco, N., Kenway, G. K. W., He, P., Mader, C. A., and Martins, J. R. R. A., "Efficient Mesh Generation and Deformation for Aerodynamic Shape Optimization," *AIAA Journal*, Vol. 59, No. 4, 2021, pp. 1151–1168. https://doi.org/10.2514/1.J059491.

[21] Wu, N., Kenway, G., Mader, C. A., Jasa, J., and Martins, J. R. R. A., "pyOptSparse: A Python framework for large-scale constrained nonlinear optimization of sparse systems," *Journal of Open Source Software*, Vol. 5, No. 54, 2020, p. 2564. https://doi.org/10.21105/joss.02564.