Aerothermal Shape Optimization of Actively-Cooled Battery Packs using Conjugate Heat Transfer

Ping He*, Christian A. Psenica†, and Lean Fang‡
Iowa State University, Ames, IA 50011

Mark K. Leader§
NASA Glenn Research Center, Cleveland, OH 44135

Thermal management for batteries is important for electric aircraft because battery temperature is critically important to vehicle safety, and it also has a direct impact on the efficiency of the battery system. Because ambient air is a readily available resource for aircraft, this paper considers an active cooling concept with forced convection of ambient air through the battery pack. Conjugate heat transfer analysis is used to solve the coupled aero-thermal problem, which consists of a finite-volume computational fluid dynamics solver for the fluid domain, and a conduction heat transfer solver for the solid domain. A mixed Neumann and Dirichlet boundary condition is developed for the fluid-solid interface, which allows the solid domain to completely submerge in the fluid domain. A gradient-based optimization method is adopted, and the discrete adjoint approach implemented in DAFoam is used to efficiently compute the gradients. The aero-thermal coupling for primal analysis and gradient computation is handled using the OpenMDAO-based MPhys framework. A constant heat source is prescribed for the battery cells, and the battery shape (design variable) is optimized to minimize cooling pump power and battery weight (composite objective function) while keeping the battery temperature below a threshold (constraint). The optimized design achieves a 44.6% and 1.5% reduction in the cooling pump power and battery weight, respectively, and the maximal temperature constraint is satisfied. This work has the potential to reduce battery-pack weight, improve performance, and reduce the weight of thermal management systems for electric vertical take-off and landing aircraft.

I. Introduction

Thermal management is essential for battery packs in electric aircraft, which lack the natural heat dissipation of conventional air-breathing engines [1]. Accumulated heat can damage batteries and other components, compromising the safety and performance of the aircraft. There are two primary methods for cooling the battery: passive and active. Passive cooling has the advantage of not requiring additional pumps or power, but it typically requires a greater surface area dedicated to heat rejection to the external environment, thereby increasing the overall weight of the battery pack. In contrast, active cooling can significantly reduce the battery pack weight, but it requires an efficient cooling system design to offset the added weight and power of the cooling pump [2]. Because ambient air is a readily available resource for aircraft, this paper considers an active cooling concept with forced convection of ambient air through the battery pack.

Conjugate heat transfer (CHT) optimization is a powerful approach to designing high-performance active cooling battery packs. It enables more realistic aero-thermal coupling by integrating computational fluid dynamics (CFD) to simulate fluid domains and conduction heat transfer solvers for the solid domains. The CHT optimization method has been used to analyze the aerodynamics and heat transfer of various battery pack shapes and flow conditions [3, 4]. CHT-based design optimization studies have also been conducted to improve

*Assistant Professor, Department of Aerospace Engineering, AIAA Senior Member. Email: phe@iastate.edu
†PhD Student, Department of Aerospace Engineering, AIAA Student Member.
‡PhD Student, Department of Aerospace Engineering, AIAA Student Member.
§Research Engineer, Propulsion Systems Analysis Branch, AIAA Member.

performance [5]; however, in this study, a gradient-free optimization algorithm was used. To allow for a relatively large design space, this paper considers a gradient-based optimization algorithm through the use of efficient adjoint-based gradient computation. However, the adjoint method requires detailed knowledge of the fluid and solid solvers (solver-intrusive), which requires time-intensive development. Gradient-based CHT optimization has been recently demonstrated using the adjoint method by Gkaragkounis et al. [6], Gallorini et al. [7], and Imam-Lawal et al. [8].

This paper is a step forward in gradient-based CHT optimization, leveraging an efficient discrete adjoint-based optimization toolbox called DAFoam* [9, 10] to consider active battery pack shape optimization with CHT. The most salient feature is that the discrete adjoint is coupled with a modular multidisciplinary design optimization (MDO) library called MPhys† [11], which allows easy extensions for more complicated optimizations, such as aero-structural-thermal optimizations. MPhys is built using OpenMDAO‡ [12], an open-source optimization framework. A heat source is prescribed for the battery cells, and the battery pack shape (design variable) is optimized to dissipate the heat. The objective function is a weighted sum of cooling pump power and battery pack weight, and there is a constraint on the maximum temperature of the battery cells. The rest of the paper will introduce our CHT optimization framework and its components, followed by a detailed discussion of the optimization results.

II. Method

A. Conjugate heat transfer optimization framework

Figure 1 shows the schematic of the CHT shape optimization framework under the OpenMDAO/MPhys [11] framework. In the preprocessing step, the fluid and solid meshes are generated, along with the free-form deformation (FFD) control points for geometry parameterization. Then, the optimizer (SNOPT [13]) passes the initial design variables (displacement of FFD points) to the geometry parameterization module, pyGeo. The open-source FFD tool, pyGeo [14] is used to parameterize the design surface geometry by embedding a set of point clouds into the prescribed FFD box, and then deforms the point cloud by moving the FFD coordinates. This process works for both structured and unstructured meshes. The FFD box is generated during the preprocessing step, and fully contains the battery cells in the fluid and solid domains. Next, the updated design surface is passed to the mesh deformation module, which deforms the volume mesh using IDWarp [15]. IDWarp uses an inverse-distance weighted algorithm to smoothly deform the volume mesh based on the deformed surface mesh, which is then passed to the aerothermal analysis component. The aerothermal analysis component solves the coupled CHT problem and sends the converged state variables to the aerothermal adjoint component. The aerothermal analysis component also computes the aerothermal objective and constraint functions (e.g., battery pack weight and maximal temperature in the battery cells). Finally, the aerothermal adjoint computes the total derivatives of the objective and constraint functions and passes them back to the optimizer, which uses the function values and derivatives to update the design variables for the next optimization iteration. This process is repeated until the optimization converges.

While many components in the above CHT optimization framework have been elaborated on in previous work [16–21], the main contribution of this paper is the development of a new aerothermal analysis component with CHT, and its details are shown in Fig. 2. The coupled aerothermal analysis starts with running the CFD (fluid) solver with an updated boundary condition (BC) at the fluid-side interface. After the CFD solver converges, the transfer variables (the near wall temperature (T_w^{fluid}) and the near wall thermal parameter $(\kappa/d)_w^{\text{fluid}}$) are extracted (refer to Section II.D on the details of the CHT boundary conditions). The above two transfer variables are then interpolated from the fluid to the solid meshes by the thermal transfer component. The FUNtoFEM [22] package is used for interpolating the thermal coupling variables. Then, the finite-volume conduction heat transfer (solid) solver is run with an updated solid BC. The solid domain transfer variables (the near wall temperature (T_w^{fluid}) and the near wall thermal parameter $(\kappa/d)_w^{\text{fluid}}$) are extracted from the solid side. These two variables

*https://dafoam.github.io

†https://github.com/OpenMDAO/mphys

‡https://github.com/OpenMDAO/OpenMDAO

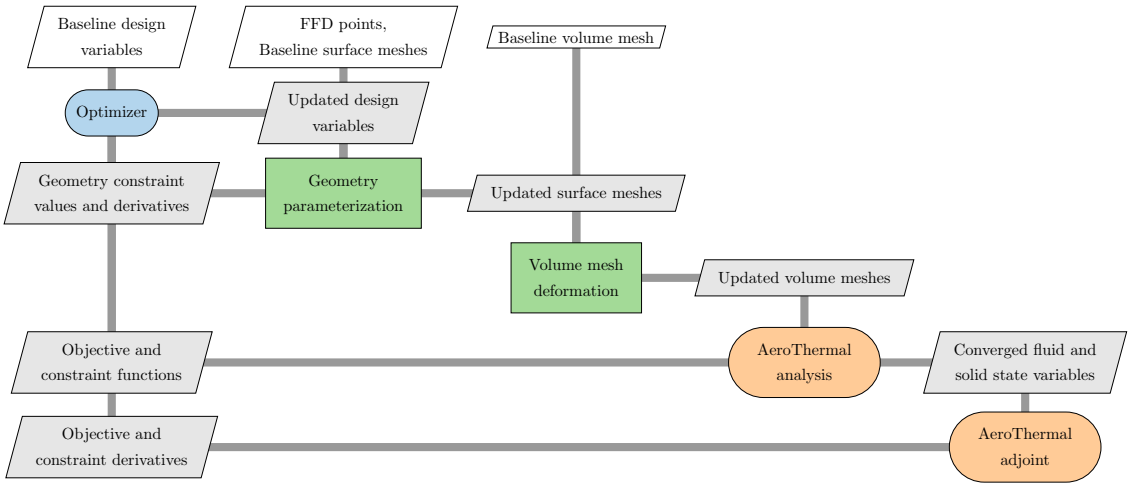

Fig. 1 Schematic of the aerothermal optimization framework.

are then interpolated back to the fluid domain for the next iteration of CFD simulations. The above process is repeated until the fluid and solid solvers' residuals are small. This paper will elaborate on the components in Fig. 2 as follows.

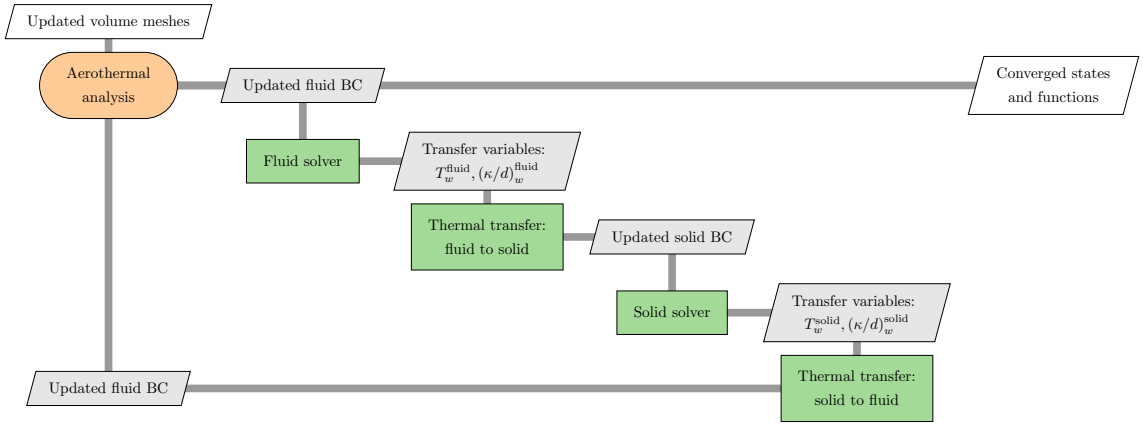

Fig. 2 Schematic of the aerothermal analysis component considering conjugate heat transfer.

B. Finite-volume CFD simulation (fluid solver component)

The primal flow analysis is conducted by using DASimpleTFoam, a modified simpleFoam solver from OpenFOAM, which adds a passive scalar equation for the temperature. DASimpleTFoam solves 3D, steady-state turbulent flows governed by the incompressible Navier-Stokes (NS) equations:

$$\nabla \cdot \boldsymbol{U} = 0 \tag{1}$$

$$\nabla \cdot (\boldsymbol{U}\boldsymbol{U}) + \nabla p - \nabla \cdot \mu_{\text{eff}}(\nabla \boldsymbol{U} + \nabla \boldsymbol{U}^T) = 0 \tag{2}$$

$$\nabla \cdot (T\boldsymbol{U}) - \alpha_{\text{eff}} \Delta T = 0 \tag{3}$$

where $\boldsymbol{U} = [u, v, w]$ is the velocity vector, p is the pressure, T is the temperature, $\mu_{\text{eff}} = \mu + \mu_t$ is the effective viscosity, i.e., the sum of dynamic viscosity and turbulent eddy viscosity, and α_{eff} is the effective thermal diffusivity.

The Spalart-Allmaras model is used to connect the turbulent eddy viscosity to the mean flow variables; the details of this formulation and description of the terms can be found in their paper [23]:

$$\nabla \cdot (\boldsymbol{U}\widetilde{v}) + \frac{1}{\sigma}\{\nabla \cdot [(v + \widetilde{v})\nabla\widetilde{v}] + C_{b1}|\nabla\widetilde{v}|^2\} - C_{b1}\widetilde{S}\widetilde{v} + C_{w1} f_w \left(\frac{\widetilde{v}}{d}\right)^2 = 0 \tag{4}$$

Then, the semi-implicit method for pressure-linked equations (SIMPLE) [24] algorithm is used to solve the incompressible NS equations. After the discretization of momentum Eq. (2), an intermediate velocity field is solved using the pressure field from the previous iteration or an initial guess, p^0.

$$a_P \boldsymbol{U}_p + \sum_N a_N \boldsymbol{U}_N = -\nabla p^0 \tag{5}$$

where a is the finite-volume discretization coefficient, the subscripts P and N denote the control volume cell and all neighboring cells, respectively. Rearranging the above equation results in:

$$a_P \boldsymbol{U}_p = -\sum_N a_N \boldsymbol{U}_N + \boldsymbol{f} - \nabla p^0 = \boldsymbol{H} - \nabla p^0 \tag{6}$$

where $\boldsymbol{H}$ represents the influence of all neighboring velocities and the source term:

$$\boldsymbol{H} = -\sum_N a_N \boldsymbol{U}_N \tag{7}$$

The face flux, ϕ, is introduced to linearize the convective term:

$$\int_S \boldsymbol{U}\boldsymbol{U} \cdot d\boldsymbol{S} = \sum_f \boldsymbol{U}_f [\boldsymbol{U}_f \cdot \boldsymbol{S}_f] = \sum_f \phi^0 \boldsymbol{U}_f \tag{8}$$

where the subscript f denotes the cell face, ϕ^0 denotes the face flux from the previous iteration or an initial guess.

An intermediate velocity field is obtained by solving the discretized momentum from Eq. (6). Note that this intermediate velocity field has not yet satisfied the continuity equation. The next step is to compute a new pressure field to correct the velocity field and face flux such that both the momentum and continuity equations are satisfied.

The discretized continuity equation is:

$$\int_S \boldsymbol{U} \cdot d\boldsymbol{S} = \sum_f [\boldsymbol{U}]_f \cdot \boldsymbol{S}_f = 0 \tag{9}$$

Here $\boldsymbol{U}_f$ is computed by interpolating the cell-centered velocity $\boldsymbol{U}_p$, which is obtained from Eq. 6, onto the cell face:

$$[\boldsymbol{U}]_f = \left(\frac{\boldsymbol{H}}{a_p}\right)_f - \left(\frac{1}{a_p}\right)_f (\nabla p)_f \tag{10}$$

This momentum interpolation method is effective in mitigating the oscillating pressure issue, as proposed by Rhie and Chow [25]. A Poisson equation is obtained by substituting Eq. (10) into Eq. (9):

$$\nabla \cdot \left(\frac{1}{a_p} \nabla p^1\right) = \nabla \cdot \left(\frac{\boldsymbol{H}}{a_p}\right) \tag{11}$$

By solving the above Poisson equation, one can obtain the updated pressure field p^1 and use it to update the face flux and the velocity field:

$$\phi^1 = [\boldsymbol{U}]_f \cdot \boldsymbol{S}_f = \left[\left(\frac{\boldsymbol{H}}{a_p} \right)_f - \left(\frac{1}{a_p} \right)_f \left(\nabla p^1 \right)_f \right] \cdot \boldsymbol{S}_f \tag{12}$$

$$\boldsymbol{U}^1 = \left(\frac{1}{a_p} \right) \left[\boldsymbol{H} - \nabla p^1 \right] \tag{13}$$

The above steps make the velocity and pressure fields satisfy both continuity and momentum equations.

Next, the passive scalar equation, Eq. (3), is solved to obtain the new temperature field, T, followed by a solution to the turbulence model equation, Eq. (4), for updating turbulence viscosity, ν_t. The above steps are repeated until all the governing equations converge.

C. Finite-volume conduction heat transfer solver (solid solver component)

A finite-volume conduction heat transfer solver is used to simulate the temperature distribution in the solid domains (battery packs). The governing equation is the Laplacian equation with a source term.

$$\kappa \nabla T_{\text{solid}} + S = 0 \tag{14}$$

where κ is the thermal conductivity, T_{solid} is the temperature in the solid domain, and S is the heat source term.

D. Conjugate heat transfer boundary condition

The interface between the fluid and solid domains needs special boundary conditions (CHT boundary conditions). As discussed in [26], using pure Dirichlet or Neumann boundary conditions may cause numerical instability or ill-defined problems. For example, the commonly used fluid-domain-fixed-temperature and solid-domain-fixed-heat-flux boundary condition is ill-posed if the solid domain is completely submerged in the fluid (like battery pack CHT problems); the solid domain would have Neumann boundary conditions for all of its surfaces. To circumvent this issue, both the fluid and solid interfaces are assigned with mixed-type (Robin) boundary conditions, similar to the OpenFOAM's `turbulentTemperatureCoupledBaffleMixed` boundary condition:

$$T_{\text{wall}} = f T_{\text{ref}} + (1 - f)(T_{\text{cell}} + d \nabla T_{\text{ref}}) \tag{15}$$

where T_{wall} is the temperature at the wall (boundary value), T_{cell} is the temperature at the mesh cell center near the wall, d is the normal distance between the near wall cell center and the wall surface, and f, T_{ref}, and ∇T_{ref} are is the user-prescribed value fraction, reference temperature, and reference temperature gradient. The following values are set: (1) $T_{\text{ref}} = T_{\text{cell}}^{\text{neighbor}}$ with the superscript neighbor being the value on the other side of the coupling interface (i.e., the neighbor domain), (2) $\nabla T_{\text{ref}} = 0$, and (3) $f = C^{\text{neighbor}}/(C^{\text{neighbor}} + C)$ with the C parameter being defined as $C^{\text{solid}} = \kappa / d^{\text{solid}}$ and $C^{\text{fluid}} = C_p \alpha_{\text{eff}} / d^{\text{fluid}}$. In summary, the interface boundary condition on the fluid side is:

$$T_{\text{wall}}^{\text{fluid}} = \frac{C^{\text{solid}}}{C^{\text{solid}} + C^{\text{fluid}}} T_{\text{cell}}^{\text{solid}} + \frac{C^{\text{fluid}}}{C^{\text{solid}} + C^{\text{fluid}}} T_{\text{cell}}^{\text{fluid}} \tag{16}$$

The interface boundary condition on the solid side is:

$$T_{\text{wall}}^{\text{solid}} = \frac{C^{\text{fluid}}}{C^{\text{solid}} + C^{\text{fluid}}} T_{\text{cell}}^{\text{fluid}} + \frac{C^{\text{solid}}}{C^{\text{solid}} + C^{\text{fluid}}} T_{\text{cell}}^{\text{solid}} \tag{17}$$

The above boundary conditions have the same right-hand side, so they ensure the wall temperatures on both sides are consistent. With further derivations, one can see that they also ensure the heat fluxes are consistent on both sides. The heat fluxes are defined as:

$$\phi^{\text{fluid}} = C_p \alpha_{\text{eff}} \frac{T^{\text{fluid}}_{\text{cell}} - T^{\text{fluid}}_{\text{wall}}}{d^{\text{fluid}}} = C^{\text{fluid}} (T^{\text{fluid}}_{\text{cell}} - T^{\text{fluid}}_{\text{wall}}) \tag{18}$$

$$\phi^{\text{solid}} = \kappa \frac{T^{\text{solid}}_{\text{cell}} - T^{\text{solid}}_{\text{wall}}}{d^{\text{solid}}} = C^{\text{solid}} (T^{\text{solid}}_{\text{cell}} - T^{\text{solid}}_{\text{wall}}) \tag{19}$$

Substituting the above Eqs. (16) and (17) into the definition of heat flux, Eqs. (18) and (19), and rearrange the terms, one can see that the heat fluxes between the fluid and solid domains are consistent with opposite signs.

$$\phi^{\text{fluid}} = \frac{C^{\text{fluid}} C^{\text{solid}} (T^{\text{fluid}}_{\text{cell}} - T^{\text{solid}}_{\text{cell}})}{C^{\text{fluid}} + C^{\text{solid}}} = -\phi^{\text{solid}} \tag{20}$$

E. Interpolation of thermal variables (thermal transfer component)

As mentioned before, the FUNtoFEM [22] package is used to interpolate the thermal coupling variables between the fluid and solid domains. FUNtoFEM uses the MELD transfer scheme, which links a fluid node with the nearest N solid nodes, and vice versa. The weights of these N nodes are computed based on their Euclidean distance. FUNtoFEM supports structured and unstructured surface meshes, with a conformal or non-conformal fluid-solid interface. In this paper, conformal surface meshes are generated for the CHT interface (the surface meshes align perfectly between the fluid and solid domains), so $N = 1$ is used.

F. Aerothermal coupled analysis and adjoint computation (aerothermal analysis component)

The combined aerothermal coupling problem, including all components described in the previous sections, is shown with more detail through OpenMDAO's N2 diagram in Fig. 3. Here the dark and blue boxes are the OpenMDAO groups and components, respectively. The dark and light green boxes are the outputs and inputs of a component, respectively. The squares in the block-diagram on the right half part illustrate the data transfer between components.

Traditionally, the aerothermal coupling is hard-coded in one solver, which makes it challenging to extend to other applications. By leveraging the flexibility of OpenMDAO and MPhys, the aerothermal coupling is implemented in a modular manner. MPhys requires users to implement the methods to compute the output based on the input for the components shown in Fig. 3, as well as the product of the state Jacobian matrix with a given vector. OpenMADO then uses the MAUD algorithm [27] to unify the adjoint total derivative computation.

To solve the coupled aerothermal analysis between the fluid and solid domains, this paper uses OpenMDAO's built-in nonlinear block Gauss–Seidel solver:

$$\begin{bmatrix} A_{\text{fluid}} & C_{\text{s2f}} \\ C_{\text{f2s}} & A_{\text{solid}} \end{bmatrix} \begin{bmatrix} w_{\text{fluid}} \\ w_{\text{solid}} \end{bmatrix} = \begin{bmatrix} b_{\text{fluid}} \\ b_{\text{solid}} \end{bmatrix} \tag{21}$$

where A and b are the nonlinear discrete matrix and vector, respectively, w are the state variable vectors, and C are the coupling variables. To ensure a robust convergence, this paper uses the Aitken under-relaxation method to update the new states for each block iteration.

To solve the coupled adjoint system, this paper uses OpenMDAO's linear block Gauss–Seidel solver:

$$\begin{bmatrix} \dfrac{\partial \boldsymbol{R}_{\text{fluid}}}{\partial \boldsymbol{w}_{\text{fluid}}}^T & \dfrac{\partial \boldsymbol{R}_{\text{solid}}}{\partial \boldsymbol{w}_{\text{fluid}}}^T \\ \dfrac{\partial \boldsymbol{R}_{\text{fluid}}}{\partial \boldsymbol{w}_{\text{solid}}}^T & \dfrac{\partial \boldsymbol{R}_{\text{solid}}}{\partial \boldsymbol{w}_{\text{solid}}}^T \end{bmatrix} \begin{bmatrix} \boldsymbol{\Psi}_{\text{fluid}} \\ \boldsymbol{\Psi}_{\text{solid}} \end{bmatrix} = \begin{bmatrix} \dfrac{\partial f}{\partial \boldsymbol{w}_{\text{fluid}}}^T \\ \dfrac{\partial f}{\partial \boldsymbol{w}_{\text{solid}}}^T \end{bmatrix} \tag{22}$$

where R is the residual, w is the state variable, and f is the objective or constraint function. DAFoam [9, 28] is used to solve the fluid and solid adjoint equations in a segregated manner with Aitken under-relaxation:

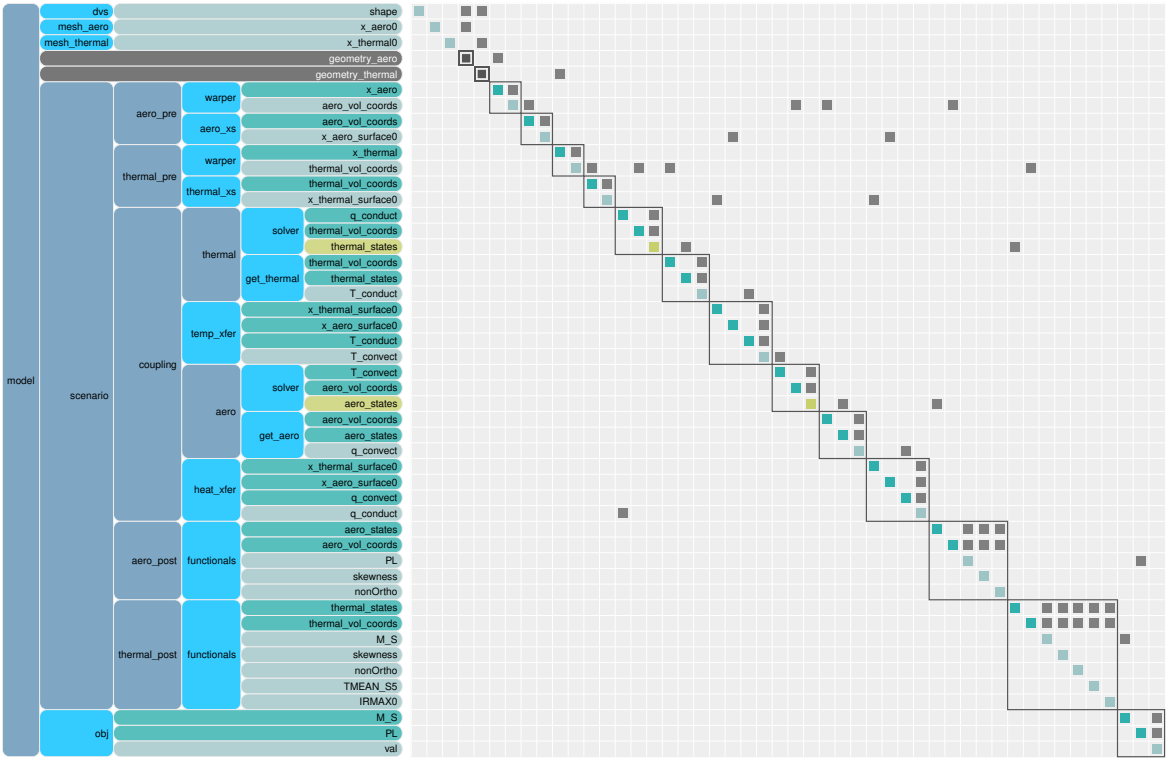

Fig. 3 N2 diagram of the aerothermal optimization framework.

$$\frac{\partial \boldsymbol{R}_{\text{fluid}}}{\partial \boldsymbol{w}_{\text{fluid}}}^{T} \boldsymbol{\Psi}_{\text{fluid}} = \frac{\partial f}{\partial \boldsymbol{w}_{\text{fluid}}}^{T} - \frac{\partial \boldsymbol{R}_{\text{solid}}}{\partial \boldsymbol{w}_{\text{fluid}}}^{T} \boldsymbol{\Psi}_{\text{solid}}, \tag{23}$$

$$\frac{\partial \boldsymbol{R}_{\text{solid}}}{\partial \boldsymbol{w}_{\text{solid}}}^{T} \boldsymbol{\Psi}_{\text{solid}} = \frac{\partial f}{\partial \boldsymbol{w}_{\text{solid}}}^{T} - \frac{\partial \boldsymbol{R}_{\text{fluid}}}{\partial \boldsymbol{w}_{\text{solid}}}^{T} \boldsymbol{\Psi}_{\text{fluid}}, \tag{24}$$

DAFoam is an open-source discrete adjoint implementation for OpenFOAM. DAFoam uses a Jacobian-free adjoint approach, in which the partial derivatives and matrix-vector products are computed using the automatic differentiation method, as detailed in Kenway et al. [29]. DAFoam uses the generalized minimal residual (GMRES), an iterative linear equation solver in the PETSc [30] library to solve the adjoint equation. A nested preconditioning strategy with the additive Schwartz method is used as the global preconditioner and the incomplete lower and upper (ILU) factorization approach with one level of fill-in for the local preconditioning. The preconditioner matrix $[\partial \boldsymbol{R}/\partial \boldsymbol{w}]_{RC}^{T}$ is constructed by approximating the residuals and their linearizations to improve convergence [10]. The construction of $[\partial \boldsymbol{R}/\partial \boldsymbol{w}]_{RC}^{T}$ is only done for the first time instance and then is reused for the adjoint equation. This treatment significantly reduces the adjoint runtime because the cost of constructing $[\partial \boldsymbol{R}/\partial \boldsymbol{w}]_{RC}^{T}$ consists of about 30% of the adjoint runtime.

III. Results and Discussion

A. Battery pack CHT problem description

This paper considers an active cooling battery pack configuration consisting of a 3×3 grid battery cells, as shown in Fig. 4. The battery cell is an 18650 lithium-ion cell with a radius of 9 mm and a height of 65 mm. The battery pack is configured such that each cell is surrounded with casing material, and the grid of cells is enclosed

on all sides by 3mm thick flat plates, with space between the cells for cooling air to pass through. The shape of the battery casing is used as the design variable for optimization to maximize performance and maintain optimal battery temperatures. The baseline design uses a cylindrical cross-sectional shape for the battery casing with a radius of 15 mm. Each battery cell is separated by 50 mm in each direction. The material for the battery casing and the top and bottom plates is aluminum.

To facilitate CFD simulations, this paper uses a simulation domain of 240 by 170 mm. This extended simulation domain in the streamwise direction artificially increases the cooling surface area. To avoid this potential issue, this paper assumes the heat generated by the battery cells is transferred across only the cylinder surfaces and a portion of the top and bottom plates, as denoted by the green squared box in Fig. 4 left and green-colored surfaces in Fig. 4 right. The rest of the top and bottom surfaces, along with the side surfaces, are adiabatic, as denoted by the grey surfaces in Fig. 4 right.

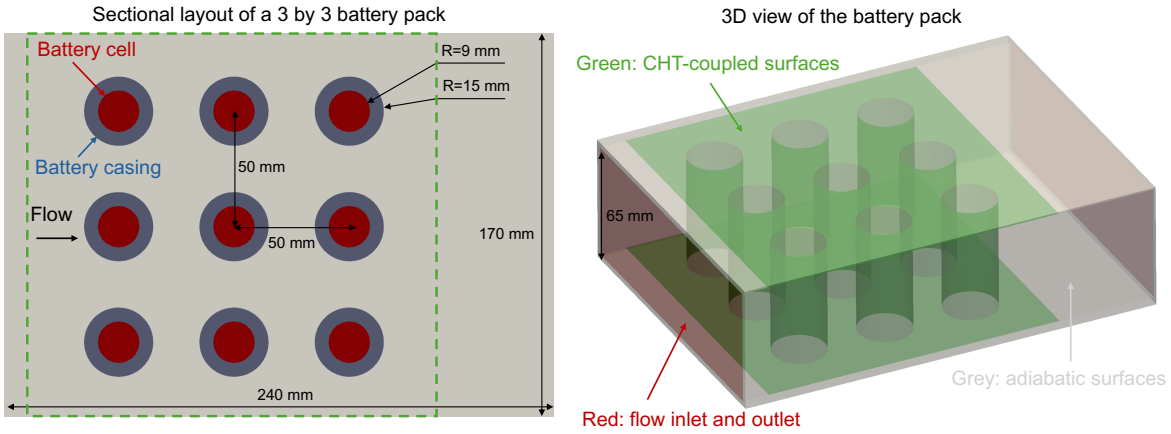

Fig. 4 Schematic of the active cooling battery pack configuration.

`Ansys Meshing` is used to generate unstructured meshes for the fluid and solid domains, as shown in Fig. 5. The fluid and solid domains have 400,000 and 320,000 mesh cells, respectively. To capture the boundary layer flow for the fluid domain, this paper uses three layers of prism mesh near all walls with an averaged y^+ of 30. The Sparlart-Allmaras turbulence model is used with a wall function. The first-order upwind discretization scheme is used for the convective term to ensure a robust steady-state CFD convergence. The above CFD settings may introduce numerical errors, compared with running unsteady flow simulations with a $y^+ \sim 1$ mesh. However, using a coarse-mesh, steady-state CFD in the CHT optimization significantly reduces the computational cost. Previous studies have demonstrated that design improvements predicted by coarse-mesh CFD can be confirmed by fine-mesh CFD simulations [31, 32]. Although the coarse mesh CFD may have errors in predicting absolute values, they often predict the correct trend for design improvement. This paper will conduct unsteady CFD with a fine mesh for the baseline and optimized designs to quantify the numerical errors from coarse-mesh CFD in Sec. III.C.

A constant heat source term is set for each battery cell in the solid domain to mimic the heat generation, as shown in Fig. 6. We use a nominal power output of the 18650 battery cell of 57.6 W, and the efficiency is assumed to be ~82%. Therefore, the heat generated by each battery cell is 10 W. The heat source has a cylindrical shape with a radius of 9 mm and a height of 65 mm, occupying the same space as the battery cell. To avoid sharp spatial variation, this paper imposes a Gaussian smooth function to decay the heat source term in the radial direction. The battery casing shape changes during optimizations; the mesh cells that occupy the battery cell space may change. Therefore, this paper re-computes and re-assigns the heat source term to the solid mesh cells for each optimization iteration; this treatment ensures the heat source term added to the solid mesh cells remains a cylindrical shape, and we ensure that the magnitude of the heat source term remains constant during optimizations.

The temperature of the incoming flow is 300 K. The maximal temperature for the battery is 328 K, and we

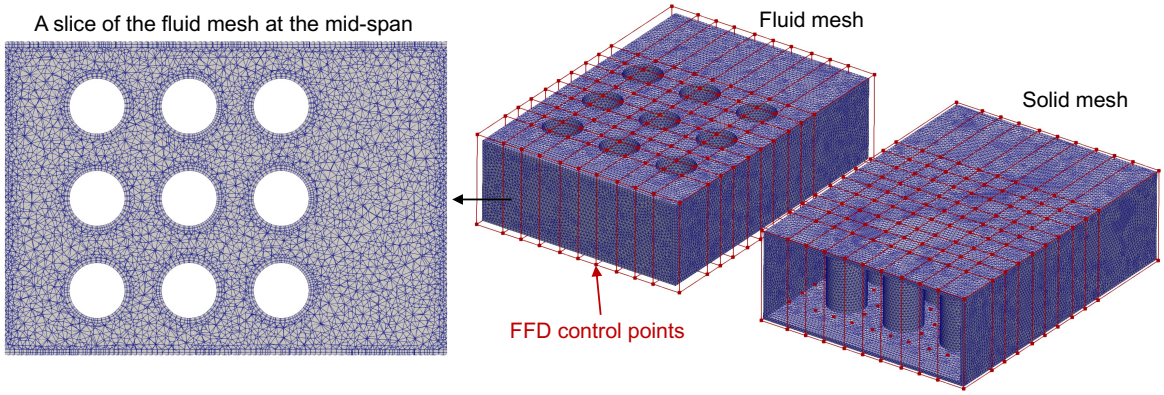

Fig. 5 Meshes for the fluid and solid domains and the free-form deformation control points (red).

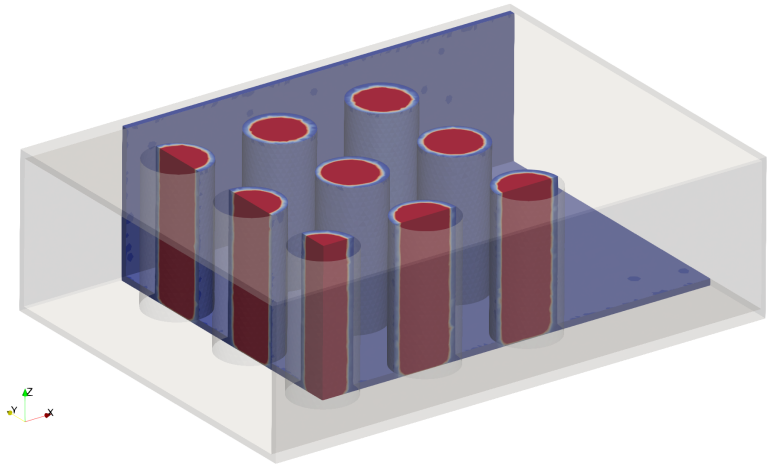

Fig. 6 Heat source term (red) added to the battery cells. Some battery cells are cut through for better illustrations.

apply a factor of safety of 1.75 based on the allowable temperature increase, resulting in a maximum temperature ($T_{\max}$) constraint of 317 K. The incoming flow speed is 7.8 m/s, which makes the baseline design feasible ($T_{\max} \simeq$ 317 K). The Reynolds number based on the battery casing diameter is 1.3×10^4.

B. Battery pack CHT optimization results and analyses

Table 1 summarizes the CHT optimization problem formulation. A composite objective function is used, which consists of pump power and battery pack weight, $f = 0.8P/P_{\text{ref}} + 0.2W/W_{\text{ref}}$, where minimizing pump power is equivalent to minimizing total pressure loss. The rationale for minimizing pump power is that this would lead to an overall more efficient system (less power required to mitigate heat), and a lighter-weight pump could be used, reducing gross takeoff weight of an aircraft. Here P is the pump power, and W is the battery weight, with the subscript "ref" denoting the baseline design reference value. The pump power is proportional to volume flow rate Q, total pressure loss Δp, and pump efficiency η. This paper assumes η and Q remain constant during optimization (constant incoming velocity). This study runs only one optimization with 80% and 20% weights; more optimizations with various other weights will be conducted in the future.

The design variable is the shape of the battery casing. The open-source pyGeo package [33] is used, which uses a free-form deformation (FFD) method to parameterize the battery casing shape. FFD control points are

Table 1 CHT optimization problem formulation.

	Function/Variable	Description	Quantity
Minimize	$f = 0.8P/P_{\text{ref}} + 0.2W/W_{\text{ref}}$	Composite function consisting of cooling pump power and battery weight	1
with respect to	$-5 \leq \Delta x \leq 5$ mm	FFD shape displacement in the x direction	4
	$-5 \leq \Delta y \leq 5$ mm	FFD shape displacement in the y direction	3
		Total Design Variables	**7**
subject to	$T_{\text{max}} < 317$ K	Max temperature constraint	1
	$\theta_{\text{fluid}} \leq 75°$	Max mesh non-orthogonality for fluid	1
	$\mu_{\text{fluid}} \leq 5$	Max mesh skewness for fluid	1
	$\theta_{\text{solid}} \leq 75°$	Max mesh non-orthogonality for fluid	1
	$\mu_{\text{solid}} \leq 5$	Max mesh skewness for fluid	1
	$r \geq 10$ mm	Min radius constraint	1
		Total Constraint Functions	**6**

generated (red points in Fig. 5 right) to fully contain the design surface, and the optimizer can then move the FFD control points to smoothly deform the design surface geometry. All battery cells are assumed to have the same shape, and their shape does not change in the spanwise (z) direction. Each battery cell is controlled by 8 FFD points, which can move in the x (streamwise) and y (lateral) directions. However, symmetry constraints are imposed in the y direction for the battery cell geometry, so the total design degrees of freedom is seven.

To ensure realistic designs, this paper imposes six geometric and physical constraints. For example, a maximal temperature constraint is imposed on the surface of the battery cells. Because the temperature is generally higher for downstream cells, only one constraint for the last column of the cells is needed. To ensure sufficient space for the battery cells inside the casing, a radius constraint is imposed such that the battery casing thickness is greater than 1 mm during optimizations. To ensure robust numerical simulations, four mesh quality constraints are set for the fluid and solid domains.

Table 2 Summary of objective functions and constraints for the baseline and optimized designs.

	Pump power	Battery weight	Max temperature
Baseline	1.000	1.000	316.93
Optimized	0.554	0.985	317.07
Difference	↓ 44.6%	↓ 1.5%	↑ 0.04%

Fig. 7 shows the optimization history for the composite objective function and temperature constraint. The optimization runs for 20 iterations. The composite objective function is reduced by 36%, and the temperature constraint is satisfied. The objective functions and constraints for the baseline and optimized designs are summarized to further evaluate the breakdown of the objective function reduction, as shown in Table 2. Both the pump power and battery weight are reduced. However, the reduction in the pump power (44.6%) is much higher than that in the battery weight. This is expected because a much higher weighting factor for the pump power is assigned in the composite objective function.

The flow fields between the baseline and optimized designs are compared to better understand the mechanism behind the design improvement. Figure 8 shows the 3D streamline (colored by the velocity magnitude) and velocity contours at different lateral locations. As shown in Fig. 8 (left), the flow accelerates through the gap between the battery cells and eventually separates, generating downstream wake vortices, which then interact

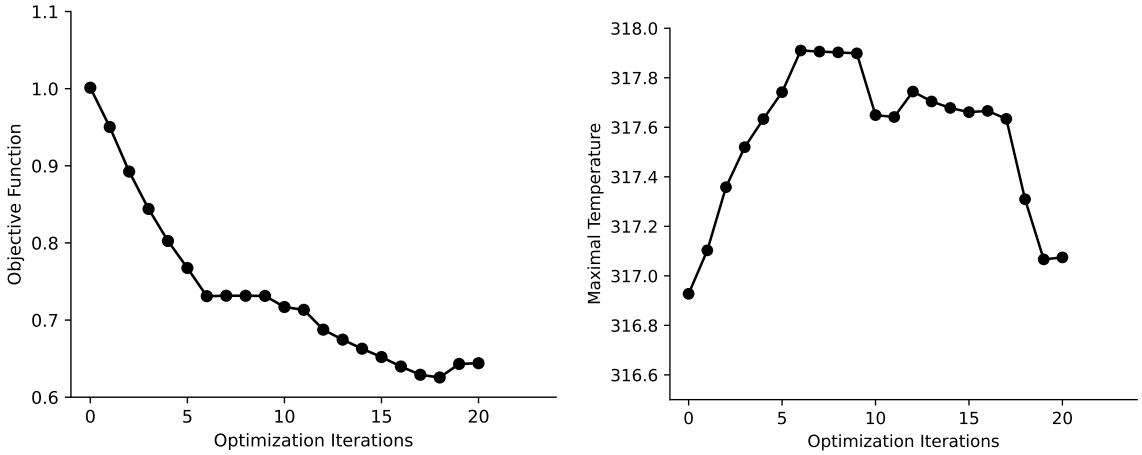

Fig. 7 Optimization history for the composite objective function (left) and the maximum temperature constraint, in Kelvin (right).

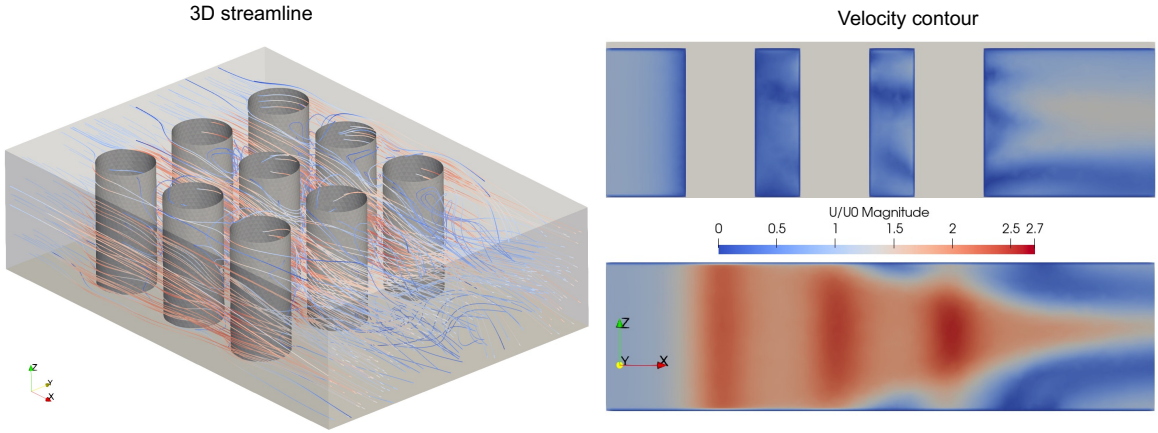

Fig. 8 Left: 3D streamline of flow over the battery pack (baseline design). Right: velocity slices at different lateral locations (top: slice cut through the battery cell center, bottom: slice between the battery cells.).

with the top and bottom walls and roll up intriguing 3D vortex structures further downstream. This can be seen more clearly in Fig. 8 (bottom-right), where low-speed vortex regions are found near the top and bottom walls.

Figure 9 shows the velocity contours at different spanwise locations for the baseline and optimized designs. For the baseline design, a relatively large velocity region downstream of the 3rd column of battery cells (midspan) is observed. However, in the same region near the bottom wall (10% span), a much lower velocity region is found; this is caused by the impact of the top and bottom walls. The optimized design creates a more streamlined battery casing geometry, which effectively reduces the size of the wake region at both spanwise locations. The smaller and weaker wakes lead to a lower total pressure loss. Note that the objective function (pressure loss and battery weight) and constraint (maximal temperatures) are competing with each other during the optimization. The optimizer tends to create a thin, streamlined battery casing geometry to reduce weight and pressure loss. However, this will reduce the cooling surface area and increase the temperature in the battery cells. Taking these into consideration, the optimizer creates the most aerodynamic and lightweight battery shape possible while keeping the battery temperature below the limit.

The above velocity distribution results in intriguing Nusselt number distributions, as shown in Fig.10. The

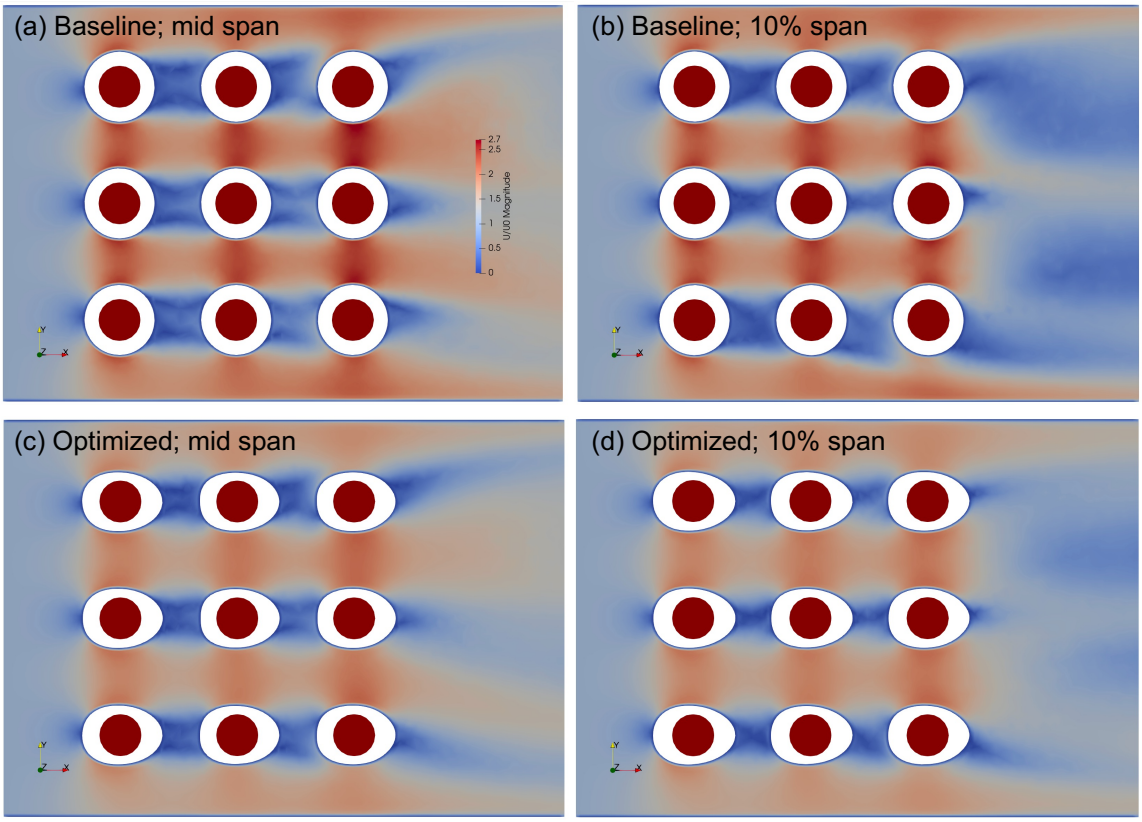

Fig. 9 Velocity contours at different spanwise locations. Left: 10% of the span. Right: 50% of the span. Top: baseline design. Bottom: Optimized design. The red circles represent the battery cells.

Nusselt number is computed as $Nu = QD/(k\Delta T)$ where Q is the heat flux per unit area, D is the battery cell diameter, k is the thermal conductivity of air, and $\Delta T = 17$ K is the reference temperature difference between the battery surface and freestream. Two high heat flux (Nusselt number) regions are observed on the battery surfaces The first is near the leading edge of the battery cylinder due to the flow impingement at a relatively high speed. The second is near the trailing edge of the battery cylinder, where the turbulence intensity is relatively high (flow separation). High Nu regions are also found on the bottom wall, which is located downstream of the battery cylinders. Again, this is caused by the strong interaction and mixing between the cylinder wake vortex and boundary layer flow near the bottom wall, as discussed above. Overall, the Nu distributions between the baseline and optimized designs are similar. This is expected because the maximal temperature constraint requires these two designs to have similar heat flux.

Figure 8 shows the temperature contours for the fluid and solid domains. Overall, there is a 1.5 K temperature difference between upstream and downstream regions. This is expected because the cooling air's temperature increases as it flows through the battery pack, lowering its capability to cool down the battery. As expected, relatively high-temperature regions near the battery cells on the top wall (Fig. 8 left) are observed, and the temperature from the upstream cells is relatively lower than that at downstream. This trend also justifies our maximal temperature constraint setup; the optimization needs to impose such a constraint for only one battery cell in the third row. The baseline and optimized designs have a similar temperature distribution; however, the temperature for the optimized design is slightly higher than the baseline. This is consistent with what is observed in Table 2 (last column).

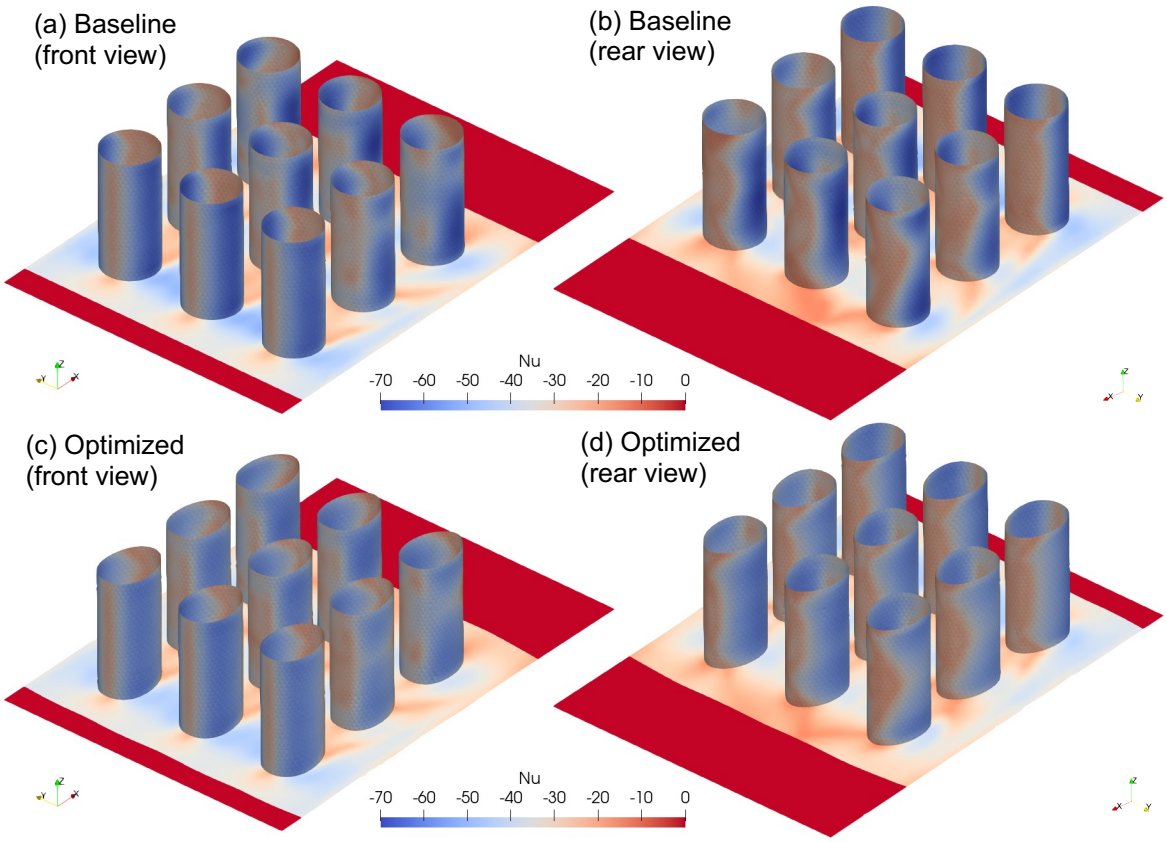

Fig. 10 The Nusselt number contours on the battery casing surfaces and bottom walls. Top: baseline design. Bottom: Optimized design. The left subfigures show a view from the front (flow entrance), and the right subfigures show a view from the rear (flow exit).

C. Error quantification for using coarse-mesh CFD in CHT optimizations

As mentioned above, this paper uses a relatively coarse mesh with $y^+ \sim 30$ to speed up the CHT optimization. In this section, fine-mesh CFD simulations are conducted to quantify to what extent the coarse-mesh CFD captures the correct trend for design improvements. Figure 12 (top) shows the 3D view of the fine mesh with 9.7 million cells, along with the y^+ contour. The average y^+ is 0.35, and the maximal y^+ is 0.96. Figure 12 (bottom) shows the 2D mesh slices at the midspan of the battery cells for the baseline and optimized designs. The pyGeo and IDWarp packages are used to deform the baseline mesh to the optimized mesh, such that these two meshes have the same mesh cells and topology. The boundary conditions are similar to the ones used in coarse-mesh simulations except for the following differences:

1) Instead of using a CHT boundary condition, a constant wall temperature condition is imposed to compute heat transfer.
2) Instead of running steady-state simulations, unsteady simulations (pimpleFoam) are conducted with a Courant number fixed to 0.8.
3) A second-order upwind scheme is used for the convective term.

Figure 13 shows the time series of drag coefficient (C_D) for the baseline and optimized designs. The baseline has a much higher time-averaged (C_D) than the optimized design, as expected. The baseline design exhibits strong flow unsteadiness due to the vortex shedding behind the battery cells. The optimized design's C_D has a much lower oscillation magnitude due to its streamlined shape. This is further illustrated by the velocity contour at the midspan shown in Fig. 14. The baseline design's instantaneous flow field (right column) differs significantly from the mean flow field (left column), while this difference is much smaller for the optimized design.

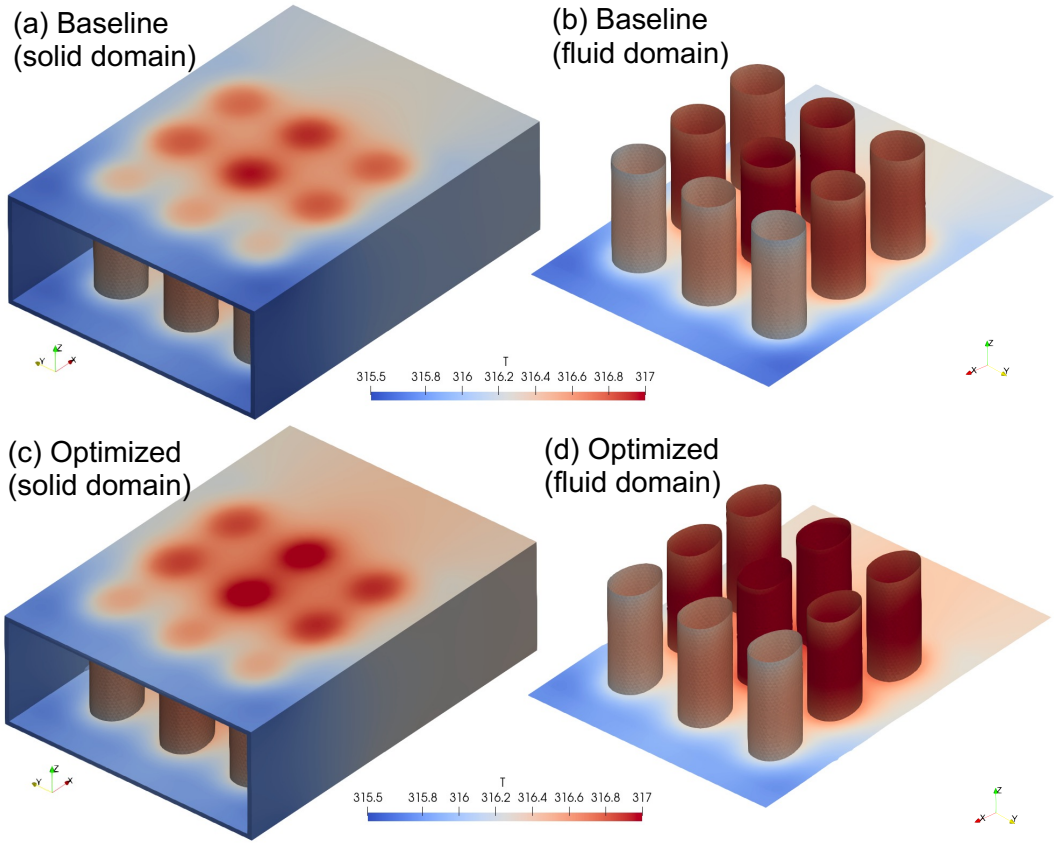

Fig. 11 Temperature contours for the fluid and solid domains. Top: baseline design. Bottom: Optimized design. The left subfigures include the pack enclosure, while the right subfigures show the cells exposed.

Table 3 Comparison of pressure loss and transferred heat for the baseline and optimized designs.

	Pressure loss (Pa)	Transferred heat (W)
Baseline (coarse mesh)	90.2	89.8
Optimized (coarse mesh)	50.0	89.6
Difference	↓ 44.6%	↓ 0.2%
Baseline (fine mesh)	105.6	107.9
Optimized (fine mesh)	48.7	95.5
Difference	↓ 53.9%	↓ 11.5%

Finally, the pressure loss and transferred heat for the baseline and optimized design are summarized in Table 3. Here both the coarse- and fine-mesh simulation results are shown. The pressure loss reduction (objective function in CHT optimizations) predicted by the fine mesh is 9.3% higher than the one predicted by the coarse mesh. This indicates that the coarse-mesh CHT optimization captures the correct trend for design improvement. On the other hand, the coarse-mesh simulation predicts that the transferred heat, which is related to the temperature constraint in CHT optimizations, remains roughly the same for the baseline and optimized designs. However, the fine-mesh simulation exhibits an 11.5% reduction in the transferred heat in the optimized design. This reduction in transferred heat will cause the battery temperature to increase, rather than remaining the same. Overall, the

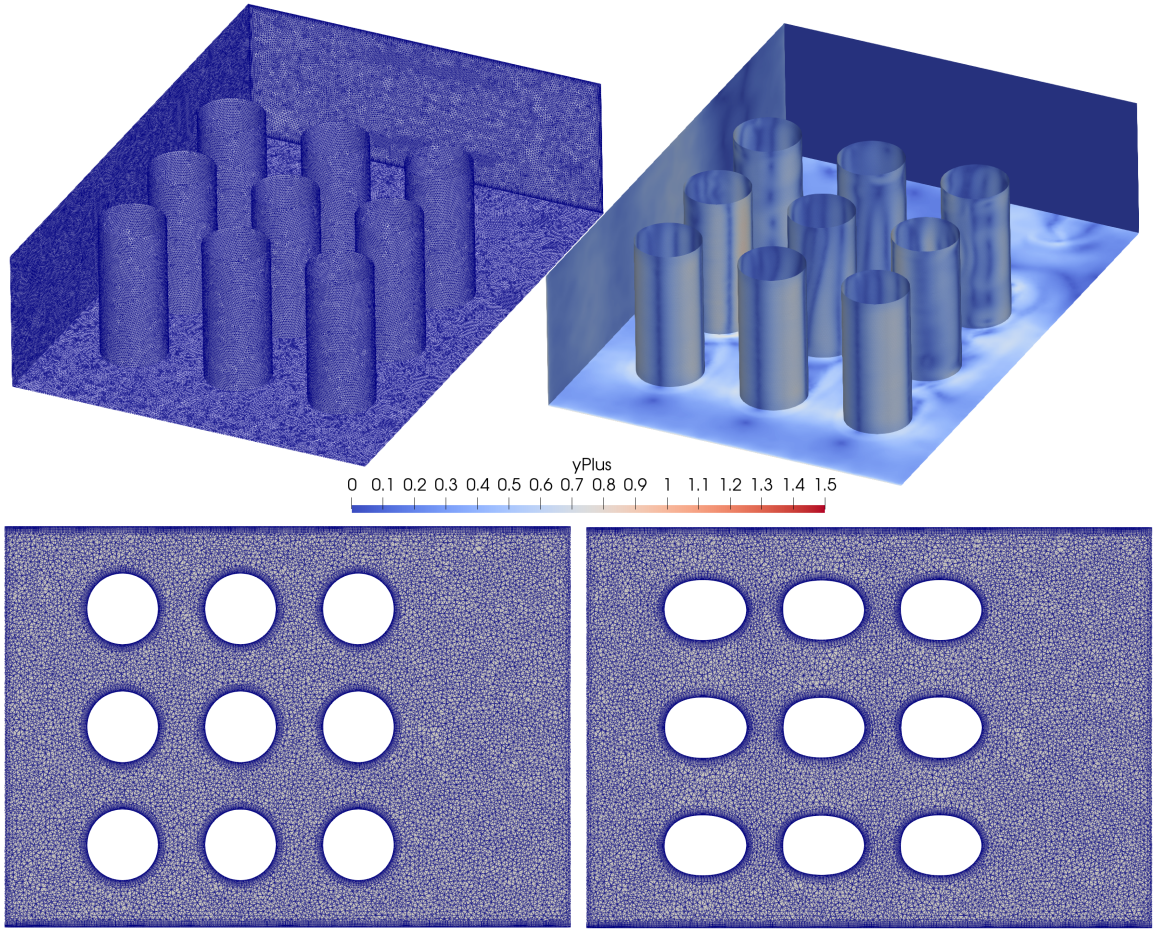

Fig. 12 Top: 3D view of the fine mesh (9.7 million cells) along with the y^+ contour. Bottom: 2D mesh slices at the midspan.

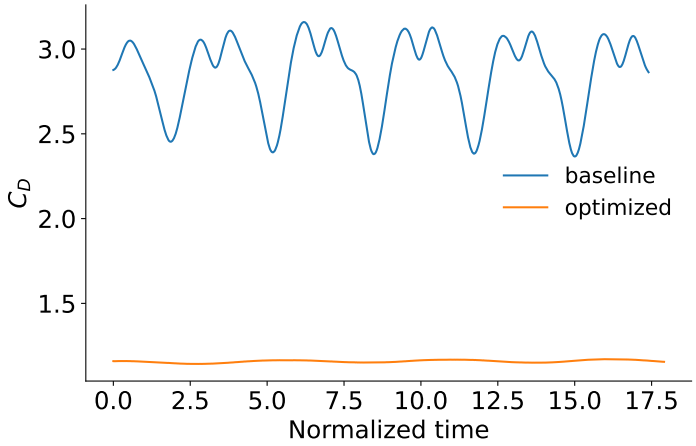

Fig. 13 Time-series of drag coefficient for the baseline and optimized designs (fine mesh).

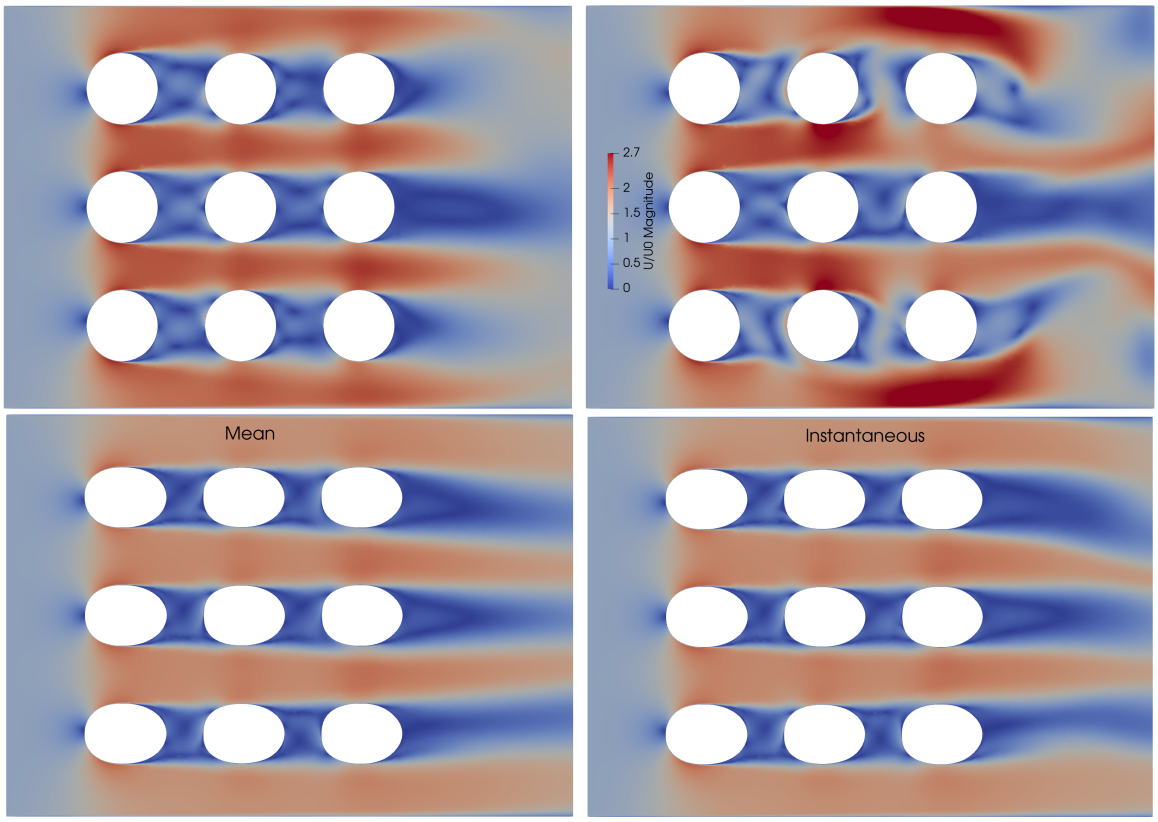

Fig. 14 Velocity contour at the midspan for the fine-mesh simulation. Top: baseline design. Bottom: optimized design.

coarse-mesh CFD has errors of approximately 10% in predicting both the pressure loss and heat transfer. This level of error is acceptable as it is within the margin of design improvement (> 40% power reduction).

IV. Summary

This paper introduces our recently developed aerothermal optimization framework based on DAFoam and OpenMDAO/MPhys and demonstrates its application for battery pack optimization. A conjugate heat transfer analysis is conducted to solve the coupled aero-thermal problem. This analysis includes a finite-volume computational fluid dynamics (CFD) solver for the fluid domain and a conduction heat transfer solver for the solid domain. A mixed Neumann-Dirichlet boundary condition at the fluid-solid interface is developed, which allows the solid domain to be fully submerged within the fluid domain. A gradient-based optimization method is used, with gradients efficiently computed via the discrete adjoint approach implemented in DAFoam. The aero-thermal coupling for both the primal analysis and gradient computation is managed using the OpenMDAO/MPhys framework. A constant heat source is applied to the battery cells, and the battery pack's shape is modified (as the design variable) to minimize cooling pump power and battery pack weight through the use of a composite objective function. At the same time, the battery temperature is required to stay below a specified threshold (constraint). The optimized design achieves simultaneous improvement in the pump power (44.6% reduction) and battery weight (1.5% reduction), and the maximal temperature constraint is satisfied. Fine-mesh unsteady CFD simulations are conducted to quantify the errors of using coarse-mesh CFD in CHT optimization. It is found that the coarse-mesh CFD has 9.3% and 11.3% errors in predicting the pressure loss and heat transfer, compared with the fine-mesh CFD results. This level of error is acceptable as it is within the margin of design improvement (44.6% power reduction), and the optimized designs remain similar. This work has the potential to

design high-performance and light-weight thermal management systems for electric vertical take-off and landing aircraft.

V. Acknowledgement

This material is based upon work supported by the National Aeronautics and Space Administration Established Program to Stimulate Competitive Research (NASA-EPSCoR) under Grant No. 80NSSC23M0159. This work used the Stampede 3 supercomputer at Texas Advanced Computing Center through allocation ATM-140019 from the Advanced Cyberinfrastructure Coordination Ecosystem: Services & Support (ACCESS) program, which is supported by U.S. National Science Foundation grants #2138259, #2138286, #2138307, #2137603, and #2138296.

References

[1] Brelje, B. J., and Martins, J. R., "Electric, hybrid, and turboelectric fixed-wing aircraft: A review of concepts, models, and design approaches," *Progress in Aerospace Sciences*, Vol. 104, No. June 2018, 2019, pp. 1–19. doi:10.1016/j.paerosci.2018.06.004, publisher: Elsevier Ltd.

[2] Qin, P., Sun, J., Yang, X., and Wang, Q., "Battery thermal management system based on the forced-air convection: A review," *ETransportation*, Vol. 7, 2021, p. 100097. doi:10.1016/j.etran.2020.100097.

[3] Chalise, D., Shah, K., Prasher, R., and Jain, A., "Conjugate heat transfer analysis of thermal management of a Li-ion battery pack," *Journal of Electrochemical Energy Conversion and Storage*, Vol. 15, No. 1, 2018, p. 011008. doi:10.1115/1.4038258.

[4] Mokashi, I., Afzal, A., Al-Mdallal, Q., Sundar, L. S., Khan, S. A., Abdullah, N. A., Azami, M. H., and Saleel, C. A., "Effect of non-conjugate and conjugate condition on heat transfer from battery pack," *Alexandria Engineering Journal*, Vol. 61, No. 4, 2022, pp. 3131–3145. doi:10.1016/j.aej.2021.08.042.

[5] Afzal, A., Samee, A. M., Jilte, R., Islam, M. T., Manokar, A. M., and Razak, K. A., "Battery thermal management: An optimization study of parallelized conjugate numerical analysis using Cuckoo search and Artificial bee colony algorithm," *International Journal of Heat and Mass Transfer*, Vol. 166, 2021, p. 120798. doi:10.1016/j.ijheatmasstransfer.2020.120798.

[6] Gkaragkounis, K. T., Papoutsis-Kiachagias, E. M., and Giannakoglou, K. C., "The continuous adjoint method for shape optimization in Conjugate Heat Transfer problems with turbulent incompressible flows," *Applied Thermal Engineering*, Vol. 140, 2018, pp. 351–362. doi:10.1016/j.applthermaleng.2018.05.054, publisher: Elsevier.

[7] Gallorini, E., Hèlie, J., and Piscaglia, F., "A multi region adjoint-based solver for topology optimization in conjugate heat transfer problems," *Computers & Fluids*, Vol. 266, 2023, p. 106042. doi:10.1016/j.compfluid.2023.106042.

[8] Imam-Lawal, O., Verstraete, T., and Müller, J.-D., "Discrete adjoint for coupled conjugate heat transfer," *Structural and Multidisciplinary Optimization*, Vol. 66, No. 5, 2023, p. 107. doi:10.1007/s00158-023-03549-1.

[9] He, P., Mader, C. A., Martins, J. R. R. A., and Maki, K. J., "DAFoam: An open-source adjoint framework for multidisciplinary design optimization with OpenFOAM," *AIAA Journal*, Vol. 58, No. 3, 2020, pp. 1304–1319. doi:10.2514/1.J058853.

[10] He, P., Mader, C. A., Martins, J. R. R. A., and Maki, K. J., "An aerodynamic design optimization framework using a discrete adjoint approach with OpenFOAM," *Computers & Fluids*, Vol. 168, 2018, pp. 285–303. doi:10.1016/j.compfluid.2018.04.012.

[11] Yildirim, A., Jacobson, K. E., Anibal, J. L., Stanford, B. K., Gray, J. S., Mader, C. A., Martins, J. R. R. A., and Kennedy, G. J., "MPhys: A Modular Multiphysics Library for Coupled Simulation and Adjoint Derivative Computation," *Structural and Multidisciplinary Optimization*, 2024. (In press).

[12] Gray, J. S., Hwang, J. T., Martins, J. R. R. A., Moore, K. T., and Naylor, B. A., "OpenMDAO: An open-source framework for multidisciplinary design, analysis, and optimization," *Structural and Multidisciplinary Optimization*, Vol. 59, No. 4, 2019, pp. 1075–1104. doi:10.1007/s00158-019-02211-z.

[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. doi:10.1137/S0036144504446096, publisher: SIAM.

[14] Kenway, G. K. W., Kennedy, G. J., and Martins, J. R. R. A., "A CAD-Free approach to high-fidelity aerostructural optimization," *13th AIAA/ISSMO Multidisciplinary Analysis Optimization Conference*, Fort Worth, TX, 2010. doi:10.2514/6.2010-9231, AIAA 2010-9231.

[15] Secco, N. R., 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. doi:10.2514/1.J059491.

[16] Pacini, B., Prajapati, M., Duraisamy, K., Martins, J. R., and He, P., "Towards mixed-fidelity aero-structural-acoustic optimization for urban air mobility vehicle design," *AIAA AVIATION 2023 Forum*, American Institute of Aeronautics and Astronautics, San Diego, CA, 2023. doi:10.2514/6.2023-3905, AIAA 2023-3905.

[17] He, P., Koyuncuoglu, H., Hu, H., Dhulipalla, A., Hu, H., and Hu, H., "High-fidelity aerodynamic and aerostructural optimization of UAV propellers using the adjoint method," *AIAA SCITECH 2023 Forum*, American Institute of Aeronautics and Astronautics, National Harbor, MD, 2023. doi:10.2514/6.2023-0531, AIAA 2023-0531.

[18] Koyuncuoglu, H., and He, P., "CFD based multi-component aerodynamic optimization for wing propeller coupling," *AIAA SCITECH 2023 Forum*, American Institute of Aeronautics and Astronautics, National Harbor, MD, 2023. doi:10.2514/6.2023-1844, AIAA 2023-1844.

[19] Koyuncuoglu, H. U., and He, P., "Simultaneous wing shape and actuator parameter optimization using the adjoint method," *Aerospace Science and Technology*, Vol. 130, 2022, p. 107876. doi:10.1016/j.ast.2022.107876, publisher: Elsevier.

[20] He, P., Martins, J. R. R. A., Mader, C. A., and Maki, K., "Aerothermal optimization of a ribbed U-Bend cooling channel using the adjoint method," *International Journal of Heat and Mass Transfer*, Vol. 140, 2019, pp. 152–172. doi:10.1016/j.ijheatmasstransfer.2019.05.075.

[21] He, P., Mader, C. A., Martins, J. R. R. A., and Maki, K. J., "Aerothermal optimization of internal cooling passages using a discrete adjoint method," *AIAA/ASME Joint Thermophysics and Heat Transfer Conference*, Atlanta, GA, 2018. doi:10.2514/6.2018-4080, AIAA 2018-4080.

[22] "FUNtoFEM code," , 2022. URL `https://github.com/smdogroup/funtofem`.

[23] Spalart, P., and Allmaras, S., "A one-equation turbulence model for aerodynamic flows," *30th Aerospace Sciences Meeting and Exhibit*, 1992. doi:10.2514/6.1992-439.

[24] Patankar, S. V., and Spalding, D. B., "A calculation procedure for heat, mass and momentum transfer in three-dimensional parabolic flows," *International Journal of Heat and Mass Transfer*, Vol. 15, No. 10, 1972, pp. 1787–1806. doi:10.1016/0017-9310(72)90054-3.

[25] Rhie, C. M., and Chow, W. L., "Numerical study of the turbulent flow past an airfoil with trailing edge separation," *AIAA Journal*, Vol. 21, No. 11, 1983, pp. 1525–1532. doi:10.2514/3.8284.

[26] Verstraete, T., and Scholl, S., "Stability analysis of partitioned methods for predicting conjugate heat transfer," *International Journal of Heat and Mass Transfer*, Vol. 101, 2016, pp. 852–869. doi:j.ijheatmasstransfer.2016.05.041.

[27] Martins, J. R. R. A., and Hwang, J. T., "Review and unification of methods for computing derivatives of multidisciplinary computational models," *AIAA Journal*, Vol. 51, No. 11, 2013, pp. 2582–2599. doi:10.2514/1.J052184.

[28] "DAFoam documentation," , 2022. URL `https://dafoam.github.io`.

[29] Kenway, G. K., Mader, C. A., He, P., and Martins, J. R., "Effective adjoint approaches for computational fluid dynamics," *Progress in Aerospace Sciences*, Vol. 110, 2019, p. 100542. doi:10.1016/j.paerosci.2019.05.002, publisher: Pergamon.

[30] Balay, S., Buschelman, K., Gropp, W. D., Kaushik, D., Knepley, M. G., McInnes, L. C., Smith, B. F., and Zhang, H., "PETSc Web page," , 2009.

[31] 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. doi:10.2514/1.J053318.

[32] Park, J., Knight, B. G., Liao, Y., Mangano, M., Pacini, B., Maki, K. J., Martins, J. R., Sun, J., and Pan, Y., "CFD-based design optimization of ducted hydrokinetic turbines," *Scientific Reports*, Vol. 13, No. 1, 2023, p. 17968. doi:10.1038/s41598-023-43724-4.

[33] Hajdik, H. M., Yildirim, A., Wu, E., Brelje, B. J., Seraj, S., Mangano, M., Anibal, J. L., Jonsson, E., Adler, E. J., Mader, C. A., Kenway, G. K. W., and Martins, J. R. R. A., "pyGeo: A geometry package for multidisciplinary design optimization," *Journal of Open Source Software*, Vol. 8, No. 87, 2023, p. 5319. doi:10.21105/joss.05319.