

The Remote Inverse Shape Design of Airfoils in Unsteady Flows

Markus P. Rumpfkeil

Ph.D. Candidate

markus@oddjob.utias.utoronto.ca

David W. Zingg

Canada Research Chair in Computational Aerodynamics

<http://goldfinger.utias.utoronto.ca/~dwz/>

University of Toronto Institute for Aerospace Studies
4925 Dufferin Street, Toronto, Ontario
Canada, M3H 5T6

Abstract

This paper uses a general framework to derive the discrete adjoint equations for the calculation of remote sensitivities in unsteady flows. These sensitivities are then successfully used in the remote inverse shape design of turbulent unsteady flow around a single-element NACA0012 airfoil as well as the remote inverse design of laminar unsteady flow around the multi-element NLR 7301 configuration, with both examples at a high angle of attack. In order to reduce the considerable computational costs associated with these remote inverse designs, the use of larger time steps over unphysical adjusting periods as well as skipping time steps while recording the flow solution are investigated and are shown to work well in practice.

Nomenclature

J	Objective function	Q	Flow variables
Y	Design variables	R	Flow residual
R^*	Unsteady flow residual	$(\nabla_Q R^*)^T$	Transpose of the unsteady flow Jacobian
Δt	Time discretization step	T	Final time
\mathcal{L}	Lagrangian	$\frac{\partial J}{\partial Y}$	Gradient of objective function
ψ	Adjoint variables	ΔT	Coarse time discretization step
N	Total number of time steps	N^*	Number of coarse time steps
p	Pressure	p^*	Target pressure

1 Introduction and Motivation

The goal of this research is to modify the shape of a high-lift airfoil configuration to minimize the radiated noise during approach while maintaining good performance. The application of numerical optimization to airframe-generated noise has not received much attention yet, but with the significant quieting of modern engines, airframe noise now competes with engine noise [1]. Thus airframe-generated noise is an important component of the total noise radiated from commercial aircraft, especially during aircraft approach and landing, when engines operate at reduced thrust, and airframe components (such as high-lift devices) are in the deployed state [2]. Future Federal Aviation Administration noise regulations, the projected growth in air travel, and the increase in population density near airports will require future civil aircraft to be substantially quieter than the current ones. Consequently, the attempt to understand and reduce airframe noise has become an important research topic [3].

This paper shows that we are able to control the near-field pressures in unsteady flows which is a major milestone towards our goal to reduce airframe-generated noise. In Section 2, we give a brief overview of how to optimize airframe-generated noise and we present the formulation of a general discrete time-dependent optimal design problem in Section 3. We then show how to apply this general formulation to remote inverse design problems in Section 4 and present several examples in Section 5. We conclude in Section 6.

2 Minimization of airframe-generated noise

Traditional adjoint implementations are aimed at minimizing a cost function computed from flow variables on a surface, for example of an airfoil, that is being modified. However, for many problems, such as noise reduction, it is desirable to be able to minimize objective functions using flow quantities that are not collocated at the points where the surface is being modified. This results in a need to quantify the influence of geometry modifications on the flow variables at an arbitrary location (e.g. a near-field plane) within the domain of interest. This type of sensitivity calculation has been successfully used by Nadarajah *et al.* [4, 5] for the steady case of sonic boom minimization and will be necessary for a variety of problems including inlet design, turbomachinery design, and our particular interest, airfoil-generated noise.

A typical approach to tackle the airfoil-generated noise reduction problem is to represent the CFD solution on a reasonable computational mesh that does not extend too far from the airfoil. At a pre-specified distance below the aircraft but still within the CFD mesh, one can specify the location of a near-field plane (see Figure 1). This plane serves as an interface between the CFD solution and a wave propagation program based on principles of geometrical acoustics and nonlinear wave propagation [6]. Such a program calculates the pressure fluctuations at a user specified ground plane which can then be used as a measure of the airframe-generated noise.

This paper only focuses on controlling the near-field pressures, which provide one of the inputs to a wave propagation program. In particular, we present the remote inverse shape design of a single-element NACA0012 airfoil at a high angle of attack in turbulent unsteady flow in Section 5.1, as well as the remote inverse design of the multi-element NLR 7301 configuration [7] at a high angle of attack in laminar unsteady flow in Section 5.2.

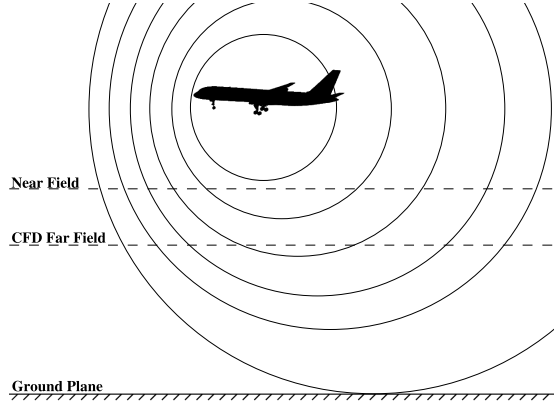


Figure 1: Schematic of the propagation of the aircraft pressure signature.

3 Formulation of the Discrete Time-dependent Optimal Design Problem

In the following we assume that we control an unsteady flow in the time interval $[0, T]$ and that we start with an initial flow solution Q^0 . Furthermore, we use the implicit Euler time marching method to discretize the governing equations in time. This is not a restriction, since it is straightforward to modify the equations to use any other time marching method (e.g. for the second-order backwards difference (BDF2) time-marching method as used in Section 5 see Appendix). The following framework was already presented in Rumpfkeil and Zingg [8] and is repeated here for completeness.

We introduce a cost function

$$J = \sum_{n=1}^N I^n(Q^n, Y), \quad (1)$$

where the function $I^n = I^n(Q^n, Y)$ depends on the time-dependent flow solution Q^n and design variables Y for $n = 1, \dots, N$. N can be calculated from the relation $T = N\Delta t$, where Δt is the chosen time discretization step. We then assume that

$$R^{*n}(Q^n, Q^{n-1}, Y) := \frac{dQ^n}{dt} + R(Q^n, Y) = \frac{Q^n - Q^{n-1}}{\Delta t} + R(Q^n, Y) = 0 \quad (2)$$

defines implicitly the time-dependent flow solution Q^n for $n = 1, \dots, N$, where $R = R(Q^n, Y)$ represents the spatially discretized convective and viscous fluxes as well as the boundary con-

ditions. We use an inexact Newton strategy to drive $R^{*n} = R^{*n}(Q^n, Q^{n-1}, Y)$ to zero [9, 10]. However, it does not matter how one solves equation (2) as long as $R^{*n} = 0$ for all n , since this is what we assume in the following.

The task of minimizing the cost function J subject to $R^{*n} = 0$ for all n can now be written as an unconstrained optimization problem of minimizing the Lagrangian function

$$\mathcal{L} = \sum_{n=1}^N [I^n(Q^n, Y) + (\psi^n)^T R^{*n}(Q^n, Q^{n-1}, Y)] \quad (3)$$

with respect to Q^0, \dots, Q^N and ψ^1, \dots, ψ^N , where ψ^1, \dots, ψ^N are the N vectors of Lagrange multipliers. A necessary condition for a minimum is that the gradient of \mathcal{L} with respect to Q^0, \dots, Q^N and ψ^1, \dots, ψ^N should be zero. Since we start with Q^0 and calculate the states Q^1, \dots, Q^N using the constraints given by equation (2), we ensure that $\nabla_{\psi^n} \mathcal{L} = 0$ for $n = 1, \dots, N$ automatically.

The Lagrange multipliers ψ^n must now be chosen such that $\nabla_{Q^n} \mathcal{L} = 0$ for $n = 1, \dots, N$, which leads to

$$0 = \nabla_{Q^n} I^n + (\psi^n)^T \nabla_{Q^n} R^{*n} + (\psi^{n+1})^T \nabla_{Q^n} R^{*n+1} \quad \text{for } n = 1, \dots, N-1 \quad (4)$$

$$0 = \nabla_{Q^N} I^N + (\psi^N)^T \nabla_{Q^N} R^{*N}. \quad (5)$$

This can be written equivalently as

$$\psi^N = -((\nabla_{Q^N} R^{*N})^T)^{-1} (\nabla_{Q^N} I^N)^T \quad (6)$$

$$\psi^n = -((\nabla_{Q^n} R^{*n})^T)^{-1} [(\nabla_{Q^n} I^n)^T + (\nabla_{Q^n} R^{*n+1})^T \psi^{n+1}] \quad \text{for } n = N-1, \dots, 1. \quad (7)$$

One can note that, since Q^1, \dots, Q^N have been calculated from the current guess of Y , that the vectors of Lagrange multipliers ψ^n can be calculated recursively backwards from the terminal boundary condition (6) using (7). The system of equations (6) and (7) is known as the system of discrete adjoint equations for the model (2), or simply as the discrete adjoint equations. In this context, the Lagrange multipliers are also known as the adjoint variables.

Finally, one can evaluate the gradient of J with respect to the design variables Y , which can then be used in a gradient-based optimization algorithm such as BFGS [11, 12, 13, 14] to find the optimum:

$$\frac{\partial J}{\partial Y} = \frac{\partial \mathcal{L}}{\partial Y} = \sum_{n=1}^N [\nabla_Y I^n(Q^n, Y) + (\psi^n)^T \nabla_Y R(Q^n, Y)]. \quad (8)$$

In summary, the gradient is determined by the solution of the adjoint equations in reverse time from the terminal boundary condition and the partial derivatives of the flow residual and objective function with respect to the design variables (while Q^n is held constant). One can also see that the computational costs of unsteady optimization problems are directly proportional to the desired number of time steps and (almost) independent of the number of design variables.

The optimal control of time-dependent problems is in general a computationally expensive task [15] since one needs to solve the adjoint equations in reverse time from a final flow solution. Thus one has to store the entire flow history, which means potentially huge memory requirements, and then to integrate the adjoint equations backwards in time which leads to equally huge processor requirements. This issue is addressed in this paper by the use of an increased time step size over unphysical adjusting periods as well as by omitting time steps while recording the flow solution as proposed by Rumpfkeil and Zingg [8].

4 Implementation of our Remote Inverse Designs

Our remote inverse design optimization procedure is as follows. First, the governing compressible two-dimensional Reynolds-averaged thin-layer Navier-Stokes equations are solved us-

ing our single-block structured solver, PROBE [10], or our multi-block structured solver, TORNADO [16]. Both solvers solve the equations in generalized coordinates using a spatial discretization based on ARC2D [17]. Time-marching is achieved through the BDF2 method and we use an inexact Newton strategy [10] implemented with the preconditioned generalized minimum residual (GMRES) method [18] and a convergence tolerance of 10^{-12} to find the root of the unsteady flow residuals R^{*n} at each time step n . The turbulent viscosity, if required, is modelled with the one-equation Spalart-Allmaras turbulence model [19].

Second, the discrete cost function J for a remote inverse design is calculated:

$$J = \frac{1}{2} \Delta t \sum_{n=N^*+1}^N \sum_{NF} (p^n - p^{*n})^2 \quad (9)$$

where p^n is the near-field pressure obtained from the current airfoil shape, and p^{*n} is the near-field target pressure obtained from a target airfoil shape (both at time step n). The sum over NF implies a sum over all the points that define the near-field plane.

Next, the adjoint equations as given in the Appendix are solved. We use the preconditioned Bi-CGSTAB algorithm [20] with an absolute convergence tolerance of 10^{-12} in order to solve the linear systems in the adjoint equations. We find that Bi-CGSTAB is about fifty percent faster than GMRES, which we use in our flow solvers. The reason for this is most likely accounted for by the fact that $(\nabla_{Q^n} R^{*n})^T$ is more diagonally dominant than the steady flow Jacobian $(\nabla_Q R)^T$ due to the extra terms on the diagonal, which makes this matrix more suited for the use of Bi-CGSTAB. However, for the unsteady flow solvers we still use the GMRES method because there are no significant computational savings for the few linear iterations we use per nonlinear (outer) iteration.

The fourth step is to calculate the gradient of J with respect to the design variables (see the Appendix). The design variables are based on a cubic B-spline parametrization of the airfoils; we use some of the vertical coordinates of the B-spline control points as design variables [21]. Furthermore, in multi-element configurations the horizontal and vertical translation of the high-lift elements can also be used as design variables.

Lastly, the BFGS optimizer [22, 23] gives us new design variable values, the airfoil geometry is updated accordingly and the grid is modified. The entire process is repeated until the gradient is sufficiently small.

5 Results

In this section we present the results of the remote inverse design problems. We consider single- and multi-element airfoils and different numbers of design variables to show that our approach works well and to be able to judge the cost of unsteady optimization.

5.1 A Single-element Airfoil in Unsteady Turbulent 2D Flow

Our first test case is a remote inverse shape design problem which involves turbulent unsteady flow over a single-element airfoil. The free stream Mach number is 0.2 with a Reynolds number of 4×10^6 , and the angle of attack is 20° . At these conditions the airfoil experiences vortex shedding. We use four shape design variables to keep the problem simple and to be able to compare our adjoint gradient with a finite-differenced one. Our initial airfoil shape is the NACA0012, and we perturb the four shape design variables slightly to get a target airfoil shape (see Figure 2).

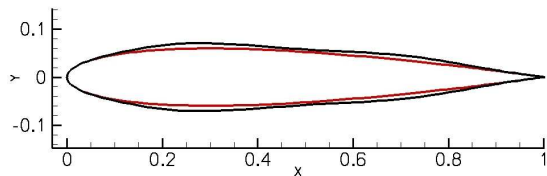


Figure 2: The initial (red) and target (black) airfoil shapes.

Our choice for the near-field plane in this case is shown in Figure 3, and the required pressures are simply obtained on the grid nodes.

In Figure 4 we show the drag coefficients for the initial and target airfoil shapes over time using a time step of $\Delta t = 0.05$. Both flow solves are warmstarted from a NACA0012 periodic steady state solution; thus one can see an adjustment period for the target airfoil. We want to “jump” over this unphysical adjusting period after a shape modification has taken place as quickly as possible. Therefore, we take a bigger time step $\Delta T = 0.1$ for the first $N^* = 300$ steps, and once we reach our desired control window, we use a smaller time step $\Delta t = 0.05$ for another 200 steps, for a total of $N = 500$ steps for each flow solve. The corresponding adjoint equations for this situation are given in the Appendix.

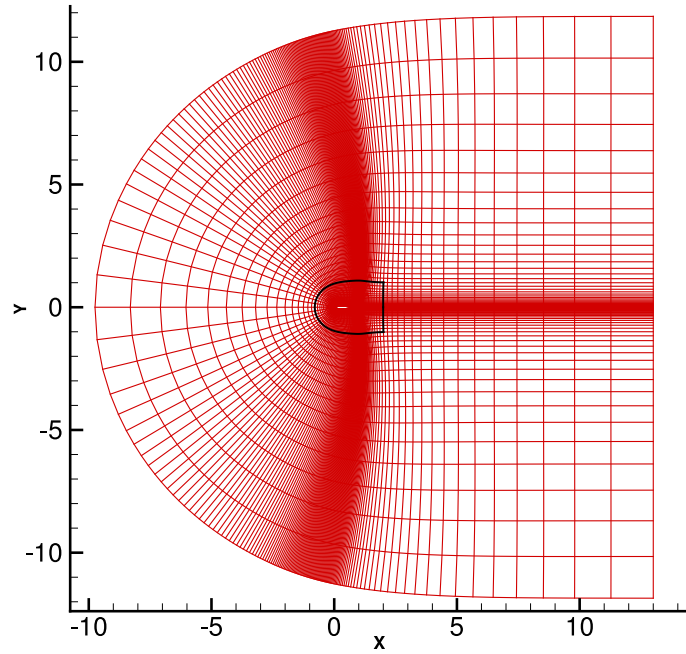


Figure 3: Our mesh; the near-field plane is shown in black.

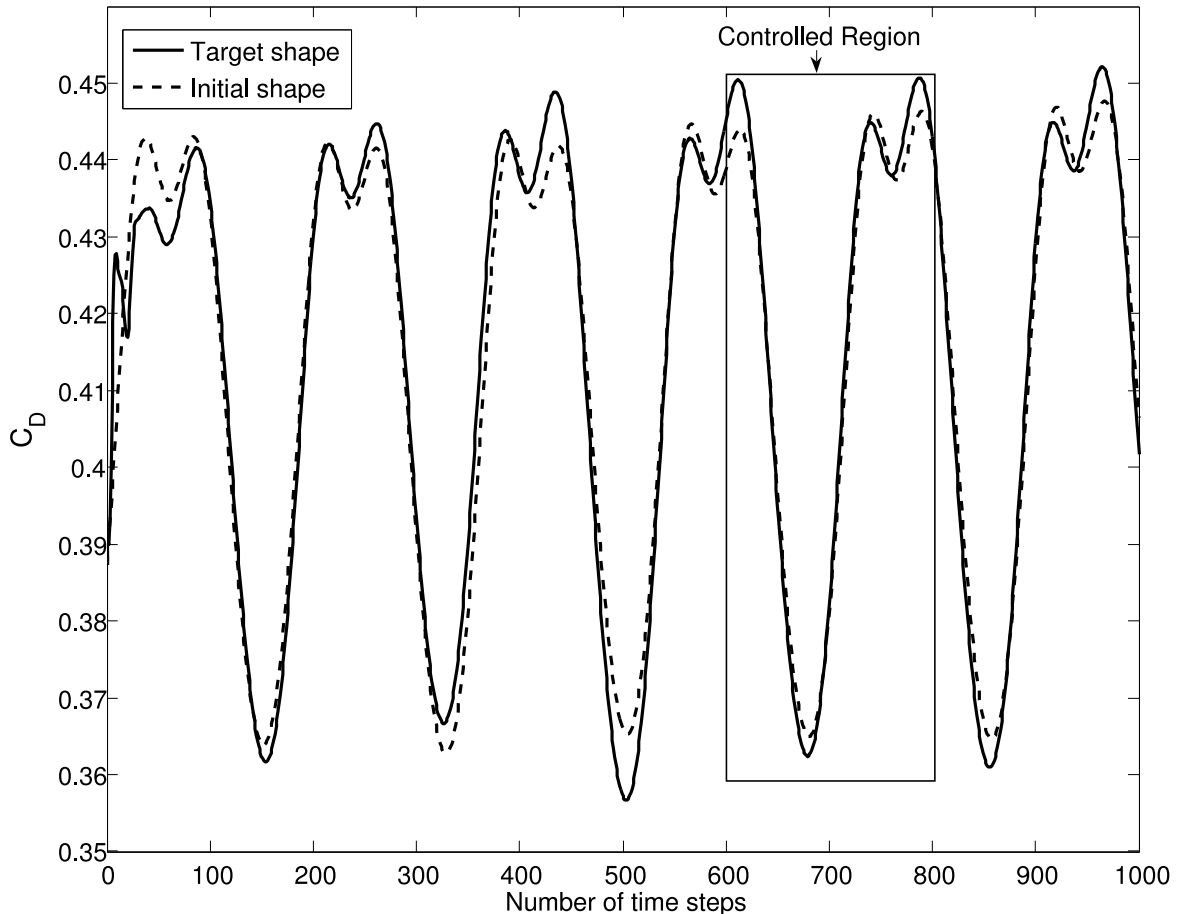


Figure 4: Drag coefficient for the initial and target airfoil shape ($\Delta t = 0.05$).

The convergence history of this remote inverse shape design problem with the adjoint approach in comparison to a second-order central finite-difference approach with a step size of $h = 10^{-7}$ is shown in Figure 5. The objective function J is always scaled such that its initial value is unity. One can see that the objective function is driven to a small value in about twenty-six design iterations and that the two approaches show a reasonable agreement, which means that our adjoint approach for the gradient calculation is accurate. We try to save computational time and storage by saving the flowfield in the adjusting period and in the control window only every fourth time step, leading to only $500/4 = 125$ matrix inversions for the solution of the adjoint equations. The result is also shown in Figure 5. The gradients and objective function values are in reasonable agreement with the original adjoint and finite-difference approach, thus leading to a similar convergence history while saving 75 percent computational resources in the adjoint calculation.

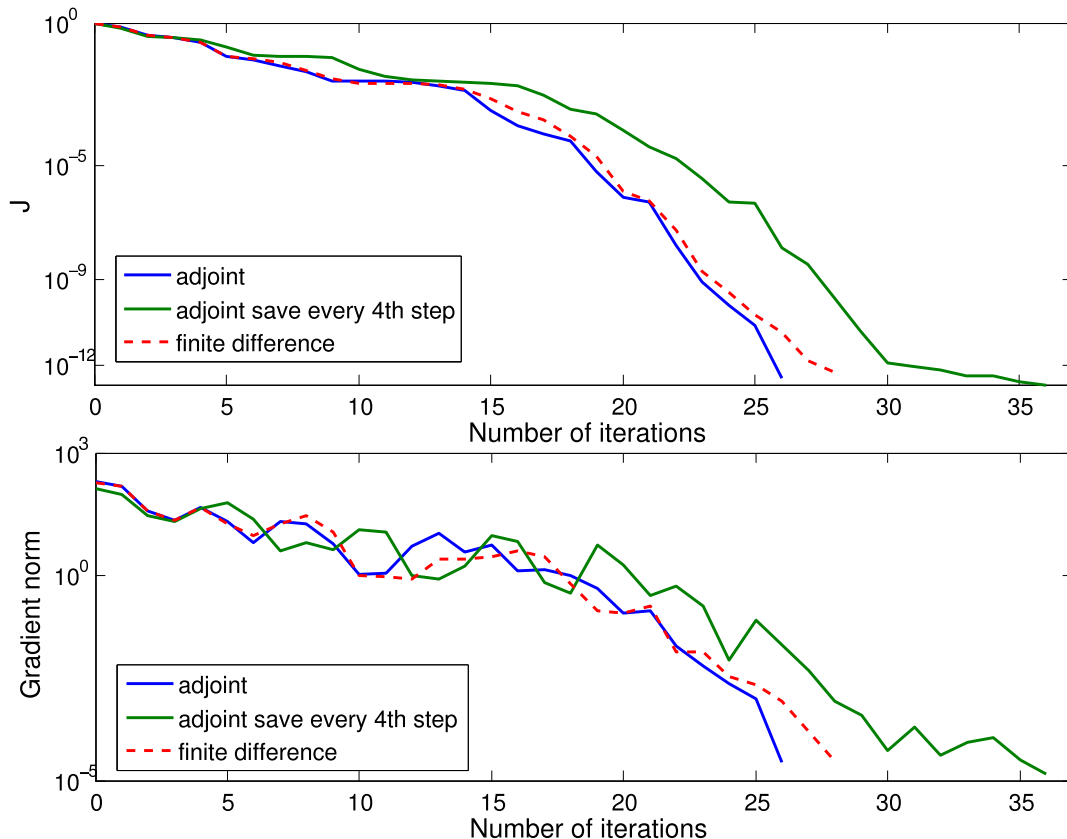


Figure 5: Convergence history of the remote inverse design problem with four design variables.

5.2 A Multi-element Airfoil in Unsteady Laminar 2D Flow

The remote inverse design problem in laminar unsteady flow over a multi-element airfoil, the NLR 7301 configuration [7], is our second test case. The free stream Mach number is 0.2 with a Reynolds number of 800, and the angle of attack is again 20° . Three cases, two with two design variables each and one with four design variables, are considered, and the shapes are displayed in Figure 6:

1. The initial airfoil is the NLR 7301, and two shape design variables of the main element are slightly perturbed to get a target airfoil.
2. The initial airfoil is the NLR 7301, and the horizontal and vertical translation design variables are slightly perturbed.
3. The initial airfoil is the NLR 7301, and two shape design variables of the main element as well as the horizontal and vertical translation design variables are slightly perturbed.

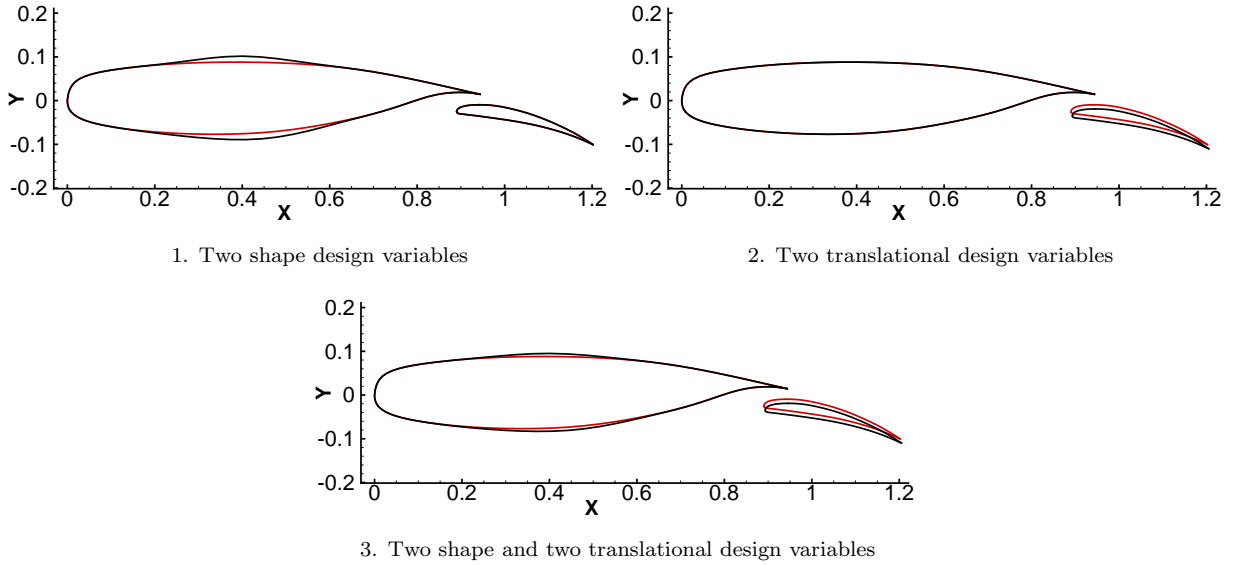


Figure 6: The initial (red) and target (black) airfoils for the three test cases.

Two different choices for the near-field plane are considered, as shown in Figure 7:

- a) The near-field plane is a square that extends from -3 to 3 with a uniform spacing of 0.05 between points in both x - and y -directions.
- b) The near-field plane is a rectangle that extends from -1 to 2 in the x -direction and from -1 to 1 in the y -direction with a uniform spacing of 0.05 between points in both directions.

The pressures in the points of the near-field plane are calculated using biquadratic interpolation involving the closest nodes of the grid to the point in question.

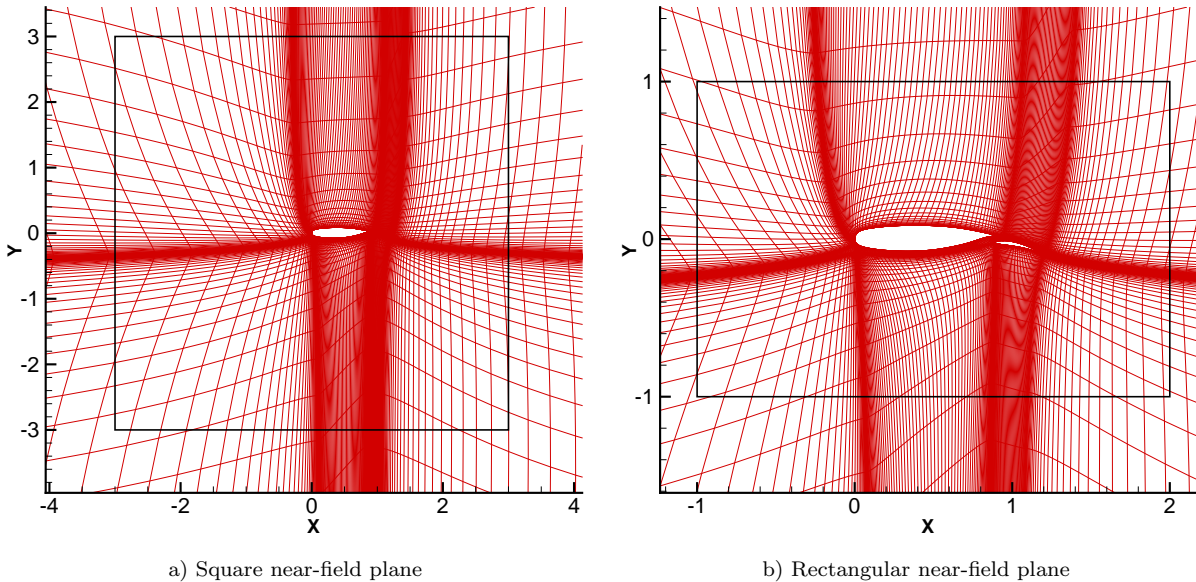


Figure 7: The grid where the two near-field planes are shown in black.

Figure 8 shows the drag coefficients for the initial and target airfoil for case 1 over time using a time step of $\Delta t = 0.1$. Once again one can see an adjustment period for the target airfoil. In order to reduce the computational costs in the actual optimization runs we “jump” over this adjusting period with a bigger time step of $\Delta T = 0.2$ for the first $N^* = 200$ steps. Once we

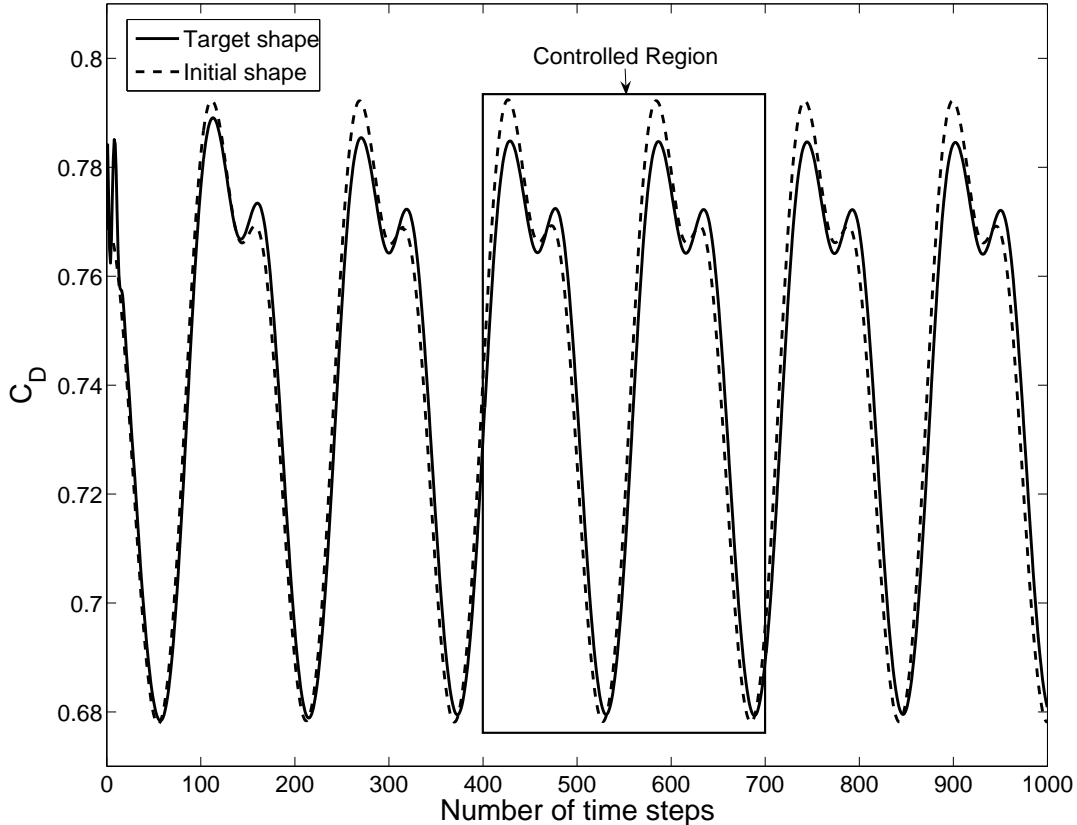


Figure 8: Drag coefficient for the initial and target airfoil for case 1 ($\Delta t = 0.1$).

reach the domain where we want to control the problem, we use a smaller time step $\Delta t = 0.1$ for another 300 steps, leading to $N = 500$ time steps in total for each flow solve.

The convergence histories of these remote inverse design problems with the adjoint approach are compared to a second-order central finite-difference approach with a step size of $h = 10^{-7}$ in Figure 9 for case 1, in Figure 10 for case 2 and in Figure 11 for the third case. The objective function J is again always scaled such that its initial value is unity. One can see that the two approaches show reasonable agreement, which means that our adjoint approach for the gradient calculation is accurate.

We also try to save computational time and storage by saving the flowfield in the adjusting period and in the control window only every fourth and even only every tenth time step for case 1, leading to only $500/4 = 125$ and $500/10 = 50$ linear solves for the solution of the adjoint equations, respectively. The result is shown in Figure 9, and the gradients and objective function values are in reasonable agreement with the original adjoint and finite-difference approach, thus leading to a somewhat similar convergence history while saving considerable computational resources.

Trying the same approach for the second case, namely saving the flowfield only every fourth and tenth time step, shows a different result (see Figure 10). This time the optimizer fails to converge if it uses only the information from every tenth time step. However, the information from only every fourth time step is still sufficient to converge in a similar manner as the original adjoint. We also try to save the flowfield only every fifth time step and one can see that this approach still works, although it comes with a huge increase in optimization iterations for the case 2a.

The third case shows yet another behaviour, as displayed in Figure 11. This time the optimizer fails to converge if it uses the information from only every tenth and every fifth time step, but the information from only every fourth time step is still sufficient to converge in a somewhat similar manner as the original adjoint.

6 Conclusion

We have presented several remote inverse design problems in unsteady turbulent and laminar flows for single- and multi-element airfoils. We showed that marching with a bigger time step over unphysical adjusting periods as well as recording the flow solution only, for example, every fourth time step works well in practice, thus resulting in significant savings in both memory and computational time for these remote inverse design problems. Our future work will focus on the ability to modify a high-lift airfoil configuration to minimize the radiated noise while maintaining good performance. Therefore, we will investigate the presented remote inverse design of a multi-element airfoil further by using more design variables, a more realistic turbulent flow and by implementing a near to far-field wave propagation method.

Acknowledgments

The funding of the second author by the Natural Sciences and Engineering Research Council of Canada and the Canada Research Chairs program is gratefully acknowledged.

Appendix

In this appendix, the discrete adjoint equations are derived in the form in which they are used to present all the results in this paper. The time-marching method of choice is the second-order accurate implicit backward difference (BDF2) method, the flow is controlled after a certain adjusting period and we can use different time step sizes in the adjusting phase and the actual control window.

The unsteady flow solve is warmstarted at some point in time which means that Q^0 and Q^{-1} are known. In order to “jump” over the adjusting period as quickly as possible, a bigger time step ΔT for N^* steps is used. Once the domain where the problem is supposed to be controlled is reached, a smaller time step Δt for another $N - N^*$ steps is used for a total of N time steps. To maintain the second-order time accuracy through this time step size change, the time-dependent flow solution Q^n is implicitly defined via the following unsteady residuals:

$$\begin{aligned}
 R^{*n}(Q^n, Q^{n-1}, Q^{n-2}, Y) &:= \frac{3Q^n - 4Q^{n-1} + Q^{n-2}}{2\Delta T} + R(Q^n, Y) = 0 \quad \text{for } n = 1, \dots, N^* \\
 R^{*N^*+1}(Q^{N^*+1}, Q^{N^*}, Q^{N^*-1}, Y) &:= \frac{(2\Delta t\Delta T + \Delta T^2)Q^{N^*+1} - (\Delta t + \Delta T)^2Q^{N^*} + \Delta t^2Q^{N^*-1}}{\Delta t\Delta T(\Delta t + \Delta T)} \\
 &\quad + R(Q^{N^*+1}, Y) = 0 \\
 R^{*n}(Q^n, Q^{n-1}, Q^{n-2}, Y) &:= \frac{3Q^n - 4Q^{n-1} + Q^{n-2}}{2\Delta t} + R(Q^n, Y) = 0 \quad \text{for } n = N^* + 2, \dots, N.
 \end{aligned}$$

The problem of minimizing the discrete objective function given by $J = \sum_{n=N^*+1}^N I^n(Q^n, Y)$ is then equivalent to the unconstrained optimization problem of minimizing the Lagrangian function

$$\mathcal{L} = \sum_{n=N^*+1}^N I^n(Q^n, Y) + \sum_{n=1}^N (\psi^n)^T R^{*n}(Q^n, Q^{n-1}, Q^{n-2}, Y)$$

with respect to Q^0, \dots, Q^N and ψ^1, \dots, ψ^N . This leads to the following equations for ψ^n :

$$\begin{aligned}
0 &= (\psi^n)^T \nabla_{Q^n} R^{*n} + (\psi^{n+1})^T \nabla_{Q^n} R^{*n+1} + (\psi^{n+2})^T \nabla_{Q^n} R^{*n+2} \\
&\quad \text{for } n = 1, \dots, N^* \\
0 &= \nabla_{Q^n} I^n + (\psi^n)^T \nabla_{Q^n} R^{*n} + (\psi^{n+1})^T \nabla_{Q^n} R^{*n+1} + (\psi^{n+2})^T \nabla_{Q^n} R^{*n+2} \\
&\quad \text{for } n = N^*+1, \dots, N-2 \\
0 &= \nabla_{Q^{N-1}} I^{N-1} + (\psi^N)^T \nabla_{Q^{N-1}} R^{*N} + (\psi^{N-1})^T \nabla_{Q^{N-1}} R^{*N-1} \\
0 &= \nabla_{Q^N} I^N + (\psi^N)^T \nabla_{Q^N} R^{*N}
\end{aligned}$$

which can be written equivalently as

$$\psi^n = \begin{cases} -((\nabla_{Q^n} R^{*n})^T)^{-1} [(\nabla_{Q^n} I^n)^T] & \text{for } n = N \\ -((\nabla_{Q^n} R^{*n})^T)^{-1} [(\nabla_{Q^n} I^n)^T + (\nabla_{Q^n} R^{*n+1})^T \psi^{n+1}] & \text{for } n = N-1 \\ -((\nabla_{Q^n} R^{*n})^T)^{-1} [(\nabla_{Q^n} I^n)^T + (\nabla_{Q^n} R^{*n+1})^T \psi^{n+1} + (\nabla_{Q^n} R^{*n+2})^T \psi^{n+2}] & \text{for } n = N-2, \dots, N^*+1 \\ -((\nabla_{Q^n} R^{*n})^T)^{-1} [(\nabla_{Q^n} R^{*n+1})^T \psi^{n+1} + (\nabla_{Q^n} R^{*n+2})^T \psi^{n+2}] & \text{for } n = N^*, \dots, 1 \end{cases}$$

A little care must be taken in calculating derivatives of R^{*N^*+1} with respect to Q^n since the factors in front of Q^{N^*+1} , Q^{N^*} and Q^{N^*-1} are slightly different. The gradient of J with respect to the design variables Y is then given by

$$\frac{\partial J}{\partial Y} = \frac{\partial \mathcal{L}}{\partial Y} = \sum_{n=N^*+1}^N \nabla_Y I^n(Q^n, Y) + \sum_{n=1}^N (\psi^n)^T \nabla_Y R(Q^n, Y).$$

References

- [1] Singer, B., Brentner, K., and Lockard, D., “Computational Aeroacoustic Analysis of Slat Trailing-Edge Flow,” *AIAA Journal*, Vol. 38, No. 9, 2000, pp. 1558–1564.
- [2] Khorrami, M., Berkman, M., and Choudhari, M., “Unsteady Flow Computations of a Slat with a Blunt Trailing Edge,” *AIAA Journal*, Vol. 38, No. 11, 2000, pp. 2050–2058.
- [3] Singer, B. and Guo, Y., “Development of Computational Aeroacoustics Tools for Airframe Noise Calculations,” *International Journal of Computational Fluid Dynamics*, Vol. 18(6), 2004, pp. 455–469.
- [4] Nadarajah, S., Jameson, A., and Alonso, J., “An Adjoint Method for the Calculation of Remote Sensitivities in Supersonic Flow,” AIAA, 2002-0261, 2002.
- [5] Nadarajah, S., Jameson, A., and Alonso, J., “An Adjoint Method for the Calculation of Remote Sensitivities in Supersonic Flow,” *International Journal of Computational Fluid Dynamics*, Vol. 20(2), 2006, pp. 61–74.
- [6] Lyrintzis, A. S., “Surface Integral Methods in Computational Aeroacoustics - From the (CFD) Near-Field to the (Acoustic) Far-Field,” *Aeroacoustics*, Vol. 2(2), 2003, pp. 95–128.
- [7] van den Berg, B., “Boundary Layer Measurements on a Two-Dimensional Wing with Flap,” National Aerospace Lab, NLR TR 79009 U, Amsterdam, Jan. 1979.
- [8] Rumpfkeil, M. and Zingg, D., “A General Framework for the Optimal Control of Unsteady Flows with Applications,” AIAA, 2007-1128, 2007.
- [9] Isono, S. and Zingg, D., “A Runge-Kutta-Newton-Krylov Algorithm for Fourth-Order Implicit Time Marching Applied to Unsteady Flows,” AIAA, 2004-0433, 2004.

- [10] Pueyo, A. and Zingg, D., “Efficient Newton-Krylov Solver for Aerodynamic Computations,” *AIAA Journal*, Vol. 36, No. 11, 1998, pp. 1991–1997.
- [11] Broyden, C., “The Convergence of a Class of Double-Rank Minimization Algorithms,” *Journal Inst. Math. Applic.*, Vol. 6, 1970, pp. 76–90.
- [12] Fletcher, R., “A New Approach to Variable Metric Algorithms,” *Computer Journal*, Vol. 13, 1970, pp. 317–322.
- [13] Goldfarb, D., “A Family of Variable Metric Updates Derived by Variational Means,” *Mathematics of Computing*, Vol. 24, 1970, pp. 23–26.
- [14] Shanno, D., “Conditioning of Quasi-Newton Methods for Function Minimization,” *Mathematics of Computing*, Vol. 24, 1970, pp. 647–656.
- [15] Nadarajah, S. and Jameson, A., “Optimal Control of Unsteady Flows using a Time Accurate Method,” AIAA, 2002-5436, 2002.
- [16] Nemec, M. and Zingg, D., “Multipoint and Multi-Objective Aerodynamic Shape Optimization,” *AIAA Journal*, Vol. 42, No. 6, 2004, pp. 1057–1065.
- [17] Pulliam, T. H., *Efficient Solution Methods for the Navier-Stokes Equations*, Lecture Notes for the Von Karman Institute For Fluid Dynamics Lecture Series, 1986.
- [18] Saad, Y. and Schultz, M., “GMRES: A Generalized Minimal Residual Algorithm for Solving Nonsymmetric Linear Systems,” *SIAM Journal on Scientific and Statistical Computing*, Vol. 7 No. 3, 1986, pp. 856–869.
- [19] Spalart, P. R. and Allmaras, S. R., “A One-Equation Turbulence Model for Aerodynamic Flows,” AIAA, 92-0439, 1992.
- [20] van der Vorst, H., “Bi-CGSTAB: A Fast and Smoothly Converging Variant of Bi-CG for the Solution of Nonsymmetric Linear Systems,” *SIAM Journal on Scientific and Statistical Computing*, Vol. 13, 1992, pp. 631–644.
- [21] Nemec, M. and Zingg, D., “Newton-Krylov Algorithm for Aerodynamic Design Using the Navier-Stokes Equations,” *AIAA Journal*, Vol. 40, No. 6, 2002, pp. 1146–1154.
- [22] Byrd, R. H., Lu, P., Nocedal, J., and Zhu, C., “A Limited Memory Algorithm for Bound Constrained Optimization,” *SIAM J. Scientific Computing* 16, Vol. 5, 1995, pp. 1190–1208.
- [23] Zhu, C., Byrd, R., Lu, P., and Nocedal, J., “L-BFGS-B: A Limited Memory FORTRAN Code for Solving Bound Constrained Optimization Problems,” Tech. Rep. NAM-11, EECS Department, Northwestern University, 1994.

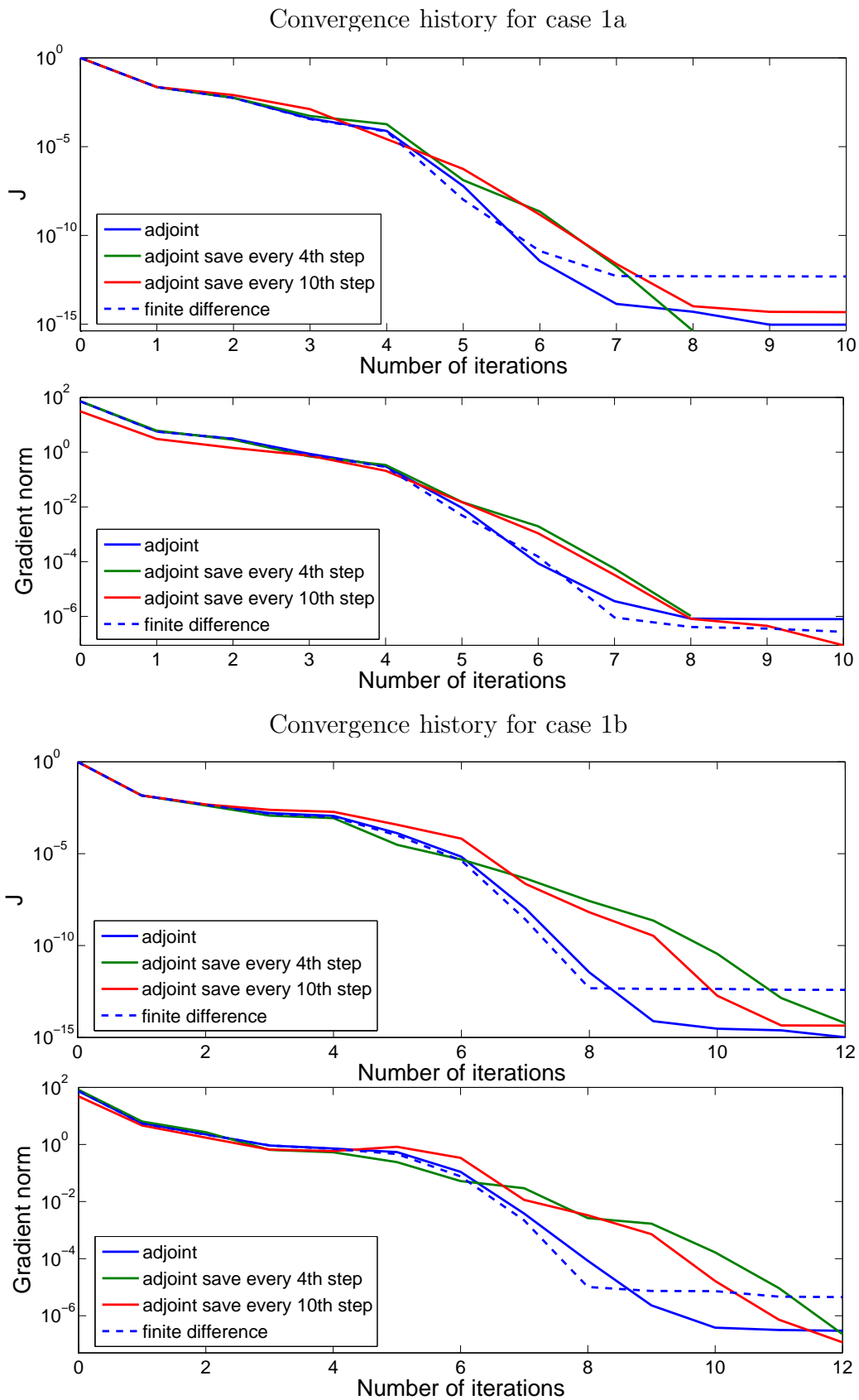


Figure 9: Convergence histories of the remote inverse design problem with two shape design variables.

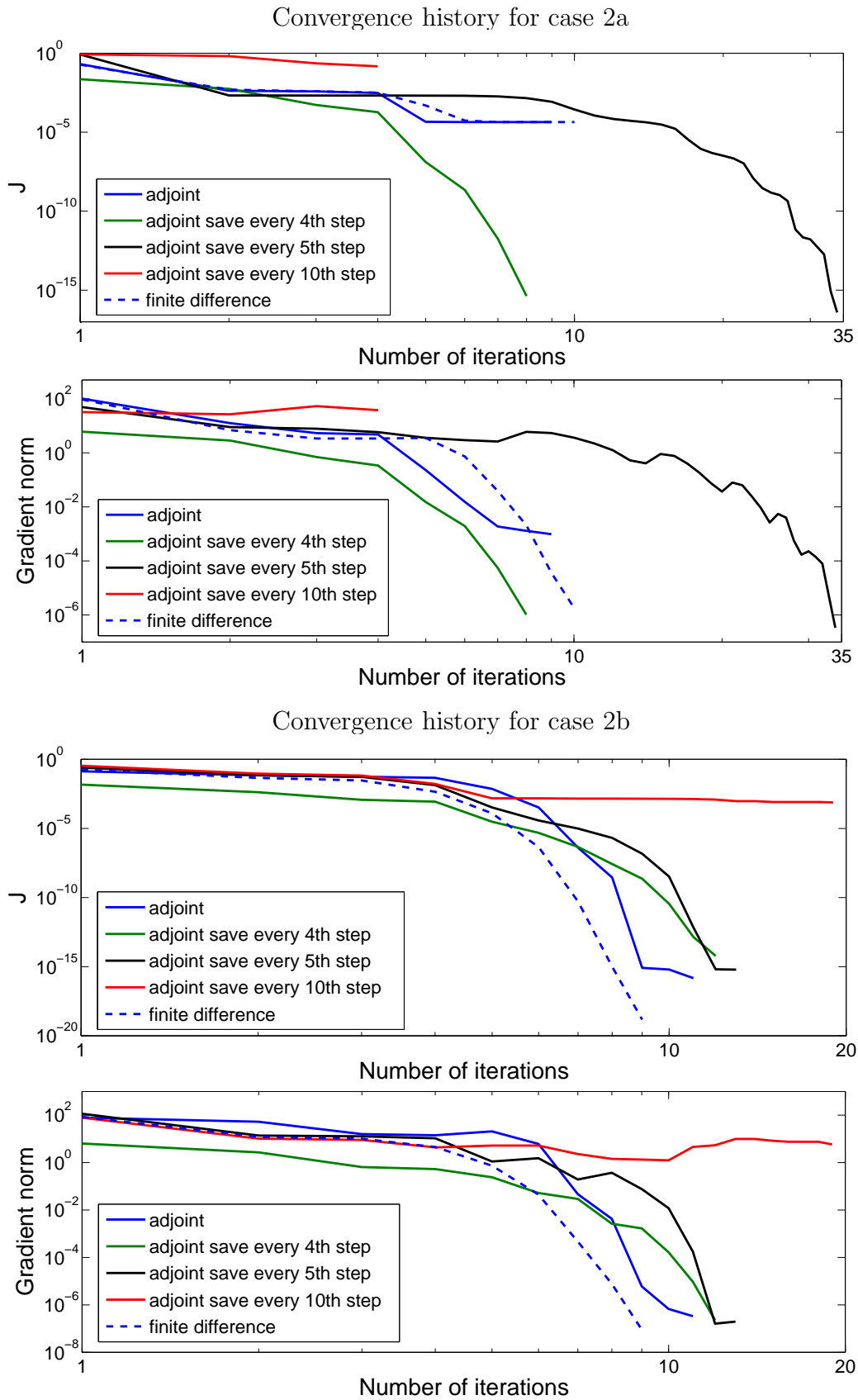


Figure 10: Convergence histories of the remote inverse design problem with two translational design variables (Note the log scale on all axes).

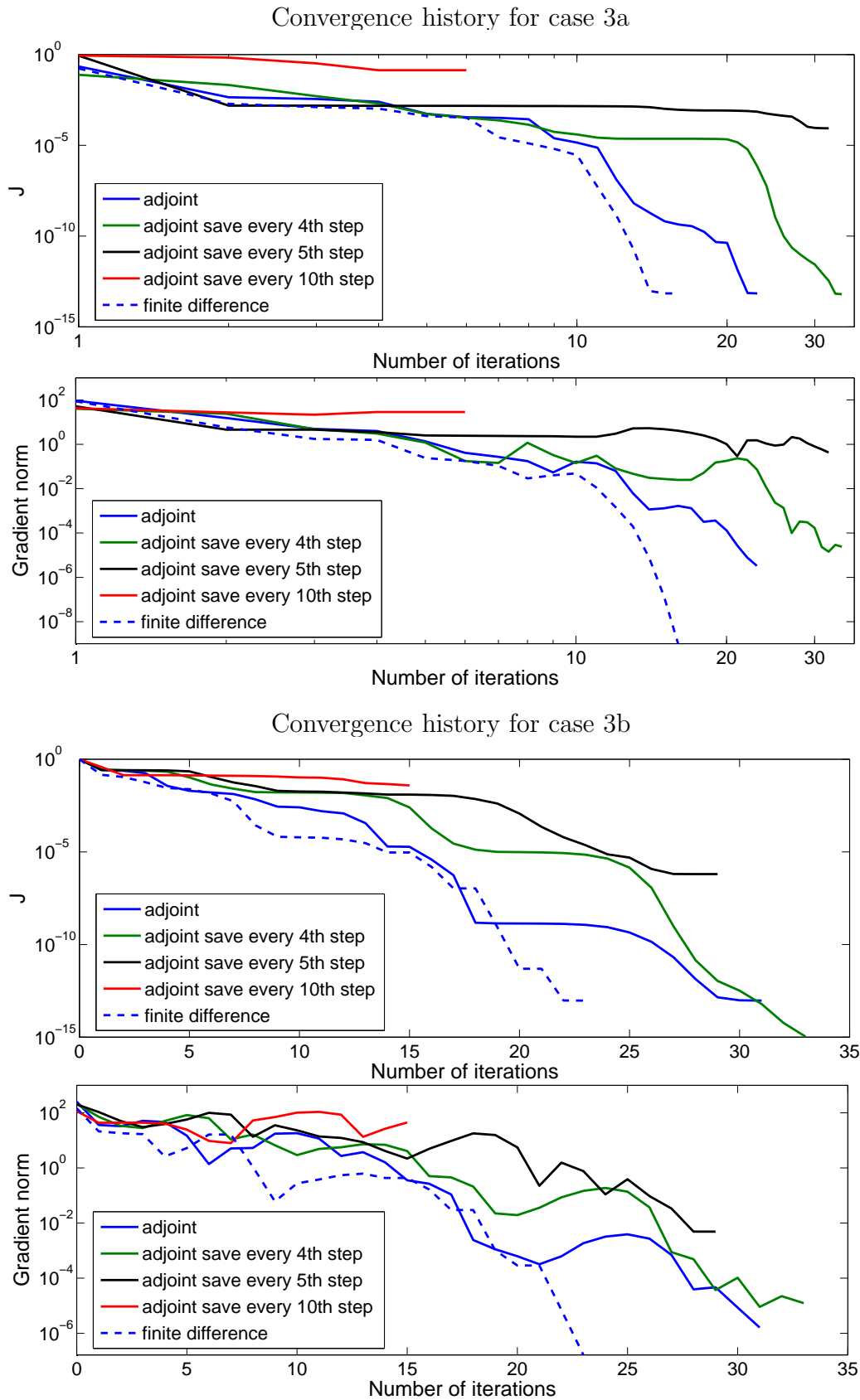


Figure 11: Convergence histories of the remote inverse design problem with two shape and two translational design variables (Note the log scale on both axes for case 3a).