47th AIAA Aerospace Sciences Meeting Including The New Horizons Forum and Aerospace Exposition
5 - 8 January 2009, Orlando, Florida

AIAA 2009-1169

Local-in-time Adjoint-based Method for Design Optimization of Unsteady Compressible Flows

N. K. Yamaleev,[*] B. Diskin,[†] and E. J. Nielsen[‡]

We develop a new local-in-time adjoint-based method for minimization of flow matching functionals subject to the 2-D unsteady compressible Euler equations. The new methodology is aimed at circumventing the memory requirements associated with conventional time-dependent adjoint-based optimization methods that require the flow solutions to be available at all time levels. These storage requirements quickly become prohibitive for large-scale applications. The key idea of the local-in-time method is to divide the entire time interval into several subintervals and approximate the solution of the global-in-time adjoint equations and the global sensitivity derivative as the combination of the corresponding local quantities computed on each subinterval. Since each subinterval contains relatively few time steps, the storage cost of the local-in-time method is much lower than that of the global adjoint formulation, thus making the time-dependent optimization feasible for practical applications. The paper presents a detailed comparison of the local- and global-in-time adjoint-based methods for design optimization of unsteady subsonic and supersonic flows around a bump. Our numerical results show that the local-in-time method converges to the optimal solution obtained with the global counterpart, while drastically reducing the memory cost as compared to the global-in-time adjoint formulation.

I. Discrete Design Optimization Problem

We consider time-dependent optimization problems that include design optimization of unsteady compressible flows governed by the 2-D Euler equations. The governing equations are discretized by using a node-centered finite-volume scheme,[1] where solution values are stored at the mesh nodes. Inviscid fluxes at cell interfaces are computed using the upwind scheme of Roe.[2] The discretized Euler equations including the boundary conditions can be written in the following form:

$$\frac{\mathbf{Q}^n - \mathbf{Q}^{n-1}}{\Delta t} + \mathbf{R}^n = \mathbf{0}, \tag{1}$$

where $\mathbf{Q} = \int_V \mathbf{U} dV$, $\mathbf{U}$ is a vector of the conserved variables, V is a control volume, $\mathbf{R}$ is the spatial undivided (by volume) flux residual, Δt is a time step, and superscript n denotes the time level. It should be noted that the above discrete formulation (1) is very general and can be directly applied to the unsteady Reynolds-averaged Navier-Stokes equations. In Eq. (1), the time derivative has been approximated using the implicit first-order backward-difference (BDF-1) formula. Note that 2nd– and 3rd-order BDF formulae can also be used in the present formulation with minor modifications.[3]

The discrete time-dependent optimization problem is formulated as follows:

$$\begin{cases} \min\limits_{\mathbf{D}\in\mathcal{D}_a} F_{\text{obj}}(\mathbf{D}), \quad F_{\text{obj}}(\mathbf{D}) = \sum\limits_{n=1}^{N} f^n(\mathbf{Q}^n, \mathbf{D})\Delta t \\ \text{subject to Eq. (1)}, \end{cases} \tag{2}$$

where $\mathbf{D}$ is a vector of the design variables, $\mathcal{D}_a$ is a set of admissible design parameters, which depends on specifics of the target physical system and ensures the existence of a solution of the optimization problem,

[*] Associate Professor, North Carolina A&T State University, Member AIAA. E-mail: nkyamale@ncat.edu

[†] Associate Fellow, National Institute of Aerospace, Member AIAA. E-mail: bdiskin@nianet.org. Also Visiting Associate Professor, MAE, University of Virginia

[‡] Research Scientist, NASA Langley Research Center, Senior Member AIAA, E-mail: Eric.J.Nielsen@nasa.gov

N is the total number of time steps, $\mathbf{Q}$ is the solution of the unsteady, compressible Euler equations (1), F_{obj} is an objective functional. The minimization problem (2) is very general and directly applicable to both active flow control and aerodynamic design optimization of unsteady flows. Similar optimal control and design optimization problems governed by the unsteady incompressible and compressible Euler/Navier-Stokes equations have been considered in.[4–11]

To reduce the complexity of the optimization problem (2), without loss of generality, it is assumed that the objective functional F_{obj} is a scalar quantity. In the present analysis, f^n in Eq. (2) is defined as follows:

$$f^n = \sum_{j \in \Gamma_c} \left[C_j^n - \left(C_j^{\text{target}} \right)^n \right]^2, \tag{3}$$

where C_j is an aerodynamic quantity such as lift or pressure coefficient on the controlled boundary surface Γ_c, C_j^{target} is a given target value of C_j. Thus, F_{obj} given by Eqs. (2, 3) is a matching-type functional.

II. Global-in-time Adjoint-based Optimization Method

The discrete time-dependent optimization problem (2) is solved by the method of Lagrange multipliers which is used to enforce the governing equations and the corresponding boundary conditions (1) as constraints. The discrete Lagrangian functional is defined as follows:

$$L(\mathbf{D}, \mathbf{Q}, \mathbf{\Lambda}) = \sum_{n=1}^{N} f^n \Delta t \quad + \sum_{n=2}^{N} \left[\mathbf{\Lambda}^n \right]^T \left(\frac{\mathbf{Q}^n - \mathbf{Q}^{n-1}}{\Delta t} + \mathbf{R}^n \right) \Delta t + \left[\mathbf{\Lambda}^1 \right]^T \left(\mathbf{Q}^1 - \mathbf{Q}^{\text{in}} \right), \tag{4}$$

where $\mathbf{\Lambda}$ is a vector of Lagrange multipliers or costate variables, $n = 1$ corresponds to the initial moment of time $t = 0$, $\mathbf{Q}^{\text{in}}$ is the initial condition for the Euler equations, f^n are given by Eq. (3), and $\mathbf{R}^n = \mathbf{R}(\mathbf{Q}^n, \mathbf{D})$ is the spatial undivided residual.

The sensitivity derivative is obtained by differentiating the Lagrangian with respect to $\mathbf{D}$, which yields

$$\begin{aligned}
\frac{dL}{d\mathbf{D}} &= \sum_{n=1}^{N} \left(\frac{\partial f^n}{\partial \mathbf{D}} + \left[\frac{\partial \mathbf{Q}^n}{\partial \mathbf{D}} \right]^T \frac{\partial f^n}{\partial \mathbf{Q}^n} \right) \Delta t + \left[\frac{\partial \mathbf{Q}^N}{\partial \mathbf{D}} \right]^T \left(\frac{\mathbf{\Lambda}^N}{\Delta t} + \left[\frac{\partial \mathbf{R}^N}{\partial \mathbf{Q}^N} \right]^T \mathbf{\Lambda}^N \right) \Delta t \\
&+ \sum_{n=2}^{N-1} \left[\frac{\partial \mathbf{Q}^n}{\partial \mathbf{D}} \right]^T \left(\frac{\mathbf{\Lambda}^n - \mathbf{\Lambda}^{n+1}}{\Delta t} + \left[\frac{\partial \mathbf{R}^n}{\partial \mathbf{Q}^n} \right]^T \mathbf{\Lambda}^n \right) \Delta t \\
&- \left[\frac{\partial \mathbf{Q}^1}{\partial \mathbf{D}} \right]^T \Lambda_2 + \left(\left[\frac{\partial \mathbf{Q}^1}{\partial \mathbf{D}} \right]^T - \left[\frac{\partial \mathbf{Q}^{\text{in}}}{\partial \mathbf{D}} \right]^T \right) \mathbf{\Lambda}^1 + \sum_{n=2}^{N} \left[\frac{\partial \mathbf{R}^n}{\partial \mathbf{D}} \right]^T \mathbf{\Lambda}^n \Delta t.
\end{aligned} \tag{5}$$

Regrouping the terms, Eq. (5) can be recast as follows:

$$\begin{aligned}
\frac{dL}{d\mathbf{D}} &= \sum_{n=1}^{N} \frac{\partial f^n}{\partial \mathbf{D}} \Delta t + \left[\frac{\partial \mathbf{Q}^N}{\partial \mathbf{D}} \right]^T \left(\frac{\mathbf{\Lambda}^N}{\Delta t} + \left[\frac{\partial \mathbf{R}^N}{\partial \mathbf{Q}^N} \right]^T \mathbf{\Lambda}^N + \frac{\partial f^N}{\partial \mathbf{Q}^N} \right) \Delta t \\
&+ \sum_{n=2}^{N-1} \left[\frac{\partial \mathbf{Q}^n}{\partial \mathbf{D}} \right]^T \left(\frac{\mathbf{\Lambda}^n - \mathbf{\Lambda}^{n+1}}{\Delta t} + \left[\frac{\partial \mathbf{R}^n}{\partial \mathbf{Q}^n} \right]^T \mathbf{\Lambda}^n + \frac{\partial f^n}{\partial \mathbf{Q}^n} \right) \Delta t \\
&+ \left[\frac{\partial \mathbf{Q}^1}{\partial \mathbf{D}} \right]^T \left(\frac{\mathbf{\Lambda}^1 - \mathbf{\Lambda}^2}{\Delta t} + \frac{\partial f^1}{\partial \mathbf{Q}^1} \right) \Delta t - \left[\frac{\partial \mathbf{Q}^{\text{in}}}{\partial \mathbf{D}} \right]^T \mathbf{\Lambda}^1 + \sum_{n=2}^{N} \left[\frac{\partial \mathbf{R}^n}{\partial \mathbf{D}} \right]^T \mathbf{\Lambda}^n \Delta t.
\end{aligned} \tag{6}$$

For aerodynamic design optimization problems, the number of design variables may be very large. Therefore, the computation of $\partial \mathbf{Q} / \partial \mathbf{D}$ is extremely expensive in terms of the CPU time, because it requires as many solves of the primary problem as the total number of the design variables involved. To eliminate this term from the Lagrangian, the second, third, and forth terms on the right hand side are set equal to zero, thus leading to the following adjoint equations for determining the Lagrange multipliers:

$$\left\{ \begin{array}{l}
\frac{1}{\Delta t} \mathbf{\Lambda}^N + \left[\frac{\partial \mathbf{R}^N}{\partial \mathbf{Q}^N} \right]^T \mathbf{\Lambda}^N = -\frac{\partial f^N}{\partial \mathbf{Q}^N}, \quad \text{for } n = N \\
\frac{1}{\Delta t} \left(\mathbf{\Lambda}^n - \mathbf{\Lambda}^{n+1} \right) + \left[\frac{\partial \mathbf{R}^n}{\partial \mathbf{Q}^n} \right]^T \mathbf{\Lambda}^n = -\frac{\partial f^n}{\partial \mathbf{Q}^n}, \quad \text{for } 2 \le n \le N-1 \\
\frac{1}{\Delta t} \left(\mathbf{\Lambda}^1 - \mathbf{\Lambda}^2 \right) = -\frac{\partial f^1}{\partial \mathbf{Q}^1}, \quad \text{for } n = 1.
\end{array} \right. \tag{7}$$

The main advantage of the adjoint formulation is that at each optimization iteration, the adjoint equations (7) are independent of $\mathbf{D}$ and should be solved once regardless of the number of the design variables. Equations (7) represent a linear system of equations for the adjoint variables, which are solved backward in time.

The matrix $\frac{\partial \mathbf{R}^n}{\partial \mathbf{Q}^n}$ and the vector $\frac{\partial f^n}{\partial \mathbf{Q}^n}$ in Eq. (7) are formed using the flow solutions $\mathbf{Q}^n$ computed and stored during the forward sweep in time. With the Lagrange multipliers found from Eq. (7), the sensitivity derivative is calculated as follows:

$$\frac{dL}{d\mathbf{D}} = \sum_{n=1}^{N} \frac{\partial f^n}{\partial \mathbf{D}} \Delta t + \sum_{n=2}^{N} \left[\frac{\partial \mathbf{R}^n}{\partial \mathbf{D}} \right]^T \mathbf{\Lambda}^n \Delta t - \left(\frac{\partial \mathbf{Q}^{\text{in}}}{\partial \mathbf{D}} \right)^T \mathbf{\Lambda}^1. \tag{8}$$

A minimum of the functional is found by the steepest descent method in which each step of the optimization cycle is taken in the negative gradient direction

$$D^l_{k+1} = D^l_k - \delta_k \left(\frac{dL}{dD^l} \right)_k, \tag{9}$$

where δ_k is an optimization step size which can be chosen adaptively, k is the optimization cycle number, D^l is an l-th component of the vector of the design variables, $\mathbf{D}$. The sensitivity derivative dL/dD^l in Eq. (9) is computed using Eq. (8) which requires the solution of the adjoint equations (7). When the adjoint equations are integrated backward in time, the sensitivity derivative at each time step is computed and added to its value at the previous time step. At $n = 1$, the complete sensitivity derivative vector is available and used in Eq. (9) for calculating a new value of each design variable D^l_{k+1}. Then, the entire optimization cycle is repeated until $\|\mathbf{D}_{k+1} - \mathbf{D}_k\| < \epsilon$, where ϵ is a given tolerance and $\|\cdot\|$ is the norm of interest. The above procedure can be summarized in the form of the following global-in-time adjoint-based algorithm:

ALGORITHM 1.

1. Choose $\mathbf{D}_1$ and set $k = 1$.
2. Solve Eq. (1) forward in time for $\mathbf{Q}^1_k, \ldots, \mathbf{Q}^N_k$ and store $\mathbf{Q}^n_k$, $n = \overline{1, N}$.
3. Solve Eq. (7) backward in time for $\mathbf{\Lambda}^n_k$, $n = \overline{N, 1}$.
4. Evaluate $\frac{dL}{d\mathbf{D}}$ using Eq. (8).
5. Choose δ_k and update D^l_{k+1} using Eq. (9) for all l.
6. If $\|\mathbf{D}_{k+1} - \mathbf{D}_k\| > \epsilon$, set $k = k + 1$ and go to step 2; otherwise stop.

III. Local-in-time Adjoint-based Optimization Method

As has been mentioned in the foregoing section, at each iteration of the global-in-time adjoint-based optimization method, the flow equations are integrated forward in time while the adjoint equations are integrated backward in time over the entire time interval considered. Since the adjoint equations (7) depend on $\mathbf{Q}^n$, the solution of the flow problem has to be stored for all time levels over which the optimization problem is solved. For realistic 3-D optimization problems, these storage requirements can quickly become prohibitive. This motivates us to study local-in-time strategies to reduce the memory cost of the global adjoint-based optimization procedure presented in the foregoing section.

We begin by dividing the entire time interval into M subintervals such that $0 = T_1 < \ldots < T_{M+1} = N\Delta t = T_{\text{final}}$, where $T_m = \Delta t N_m$, $M \le N$, and Δt is a constant time step used for integrating the flow and adjoint equations. In general, this partitioning can be chosen so that each subinterval contains one or several time steps of the time-marching scheme used for solving the flow equations. The main idea of the proposed strategy is based on the observation that the sensitivity derivative of the global-in-time Lagrangian can be represented as a sum of the local sensitivity derivatives defined at each subinterval:

$$\frac{dL}{d\mathbf{D}} = \sum_{m=1}^{M} \frac{dL^m}{d\mathbf{D}}, \tag{10}$$

where the local Lagrangian functionals are defined as

$$L^m = \begin{cases} \sum\limits_{n=N_m+1}^{N_{m+1}} f^n \Delta t + \sum\limits_{n=N_m+1}^{N_{m+1}} [\mathbf{\Lambda}^n]^T \left(\frac{\mathbf{Q}^n - \mathbf{Q}^{n-1}}{\Delta t} + \mathbf{R}^n \right) \Delta t, & \text{for } 2 \le m \le M \\ \sum\limits_{n=1}^{N_2} f^n \Delta t + \sum\limits_{n=2}^{N_2} [\mathbf{\Lambda}^n]^T \left(\frac{\mathbf{Q}^n - \mathbf{Q}^{n-1}}{\Delta t} + \mathbf{R}^n \right) \Delta t + \left[\mathbf{\Lambda}^1\right]^T \left(\mathbf{Q}^1 - \mathbf{Q}^{\text{in}} \right), & \text{for } m = 1. \end{cases} \tag{11}$$

In Eq. (11), time levels N_m and N_{m+1} correspond to T_m and T_{m+1}, respectively.

The adjoint equations corresponding to the local Lagrangian functionals, L^m for $m = \overline{1,M}$, can be derived by using the same adjoint-based approach described in the foregoing section. Differentiating each local Lagrangian, L^m, with respect to $\mathbf{D}$ and taking into account the contribution from L^{m+1} yield the following flow adjoint equations on the subinterval $(T_m, T_{m+1}]$:

$$\begin{cases} \frac{1}{\Delta t}\left(\mathbf{\Lambda}^{N_{m+1}} - \mathbf{\Lambda}^{N_{m+1}+1}\right) + \left[\frac{\partial \mathbf{R}^{N_{m+1}}}{\partial \mathbf{Q}^{N_{m+1}}}\right]^T \mathbf{\Lambda}^{N_{m+1}} = -\frac{\partial f^{N_{m+1}}}{\partial \mathbf{Q}^{N_{m+1}}} \quad \text{for } n = N_{m+1} \\ \frac{1}{\Delta t}\left(\mathbf{\Lambda}^{n} - \mathbf{\Lambda}^{n+1}\right) + \left[\frac{\partial \mathbf{R}^{n}}{\partial \mathbf{Q}^{n}}\right]^T \mathbf{\Lambda}^{n} = -\frac{\partial f^{n}}{\partial \mathbf{Q}^{n}}, \quad \text{for } N_m + 1 \le n \le N_{m+1} - 1 \\ \frac{1}{\Delta t}\left(\mathbf{\Lambda}^{1} - \mathbf{\Lambda}^{2}\right) = -\frac{\partial f^{1}}{\partial \mathbf{Q}^{1}}, \quad \text{for } n = 1, \end{cases} \tag{12}$$

where $\mathbf{\Lambda}^n$ is the solution of the global-in-time adjoint solutions defined for $N_m < n \le N_{m+1}$. In fact, the equations (11) and (12) for $m = \overline{1,M}$ are equivalent to the original equations (4) and (7), respectively. The presence of the $\mathbf{\Lambda}^{N_{m+1}+1}$ term in Eq. (13) indicates that the system of adjoint equations on the subinterval $(T_m, T_{m+1}]$ is coupled with the one defined on the next subinterval $(T_{m+1}, T_{m+2}]$ and so on. As a result, Eq. (12) for $m = \overline{1,M}$ represents a system of coupled adjoint equations on the entire time interval $[0, T_{\text{final}}]$, which implies that the entire flow solution history for all time levels has to be available when these adjoint equations are integrated backward in time.

To overcome this problem and drastically reduce the storage requirements, we approximate Eq. (13) by using the following local-in-time adjoint equations defined on the subinterval $(T_m, T_{m+1}]$:

$$\begin{cases} \frac{1}{\Delta t}\left(\hat{\mathbf{\Lambda}}^{N_{m+1}} - \tilde{\mathbf{\Lambda}}\right) + \left[\frac{\partial \mathbf{R}^{N_{m+1}}}{\partial \mathbf{Q}^{N_{m+1}}}\right]^T \hat{\mathbf{\Lambda}}^{N_{m+1}} = -\frac{\partial f^{N_{m+1}}}{\partial \mathbf{Q}^{N_{m+1}}} \quad \text{for } n = N_{m+1} \\ \frac{1}{\Delta t}\left(\hat{\mathbf{\Lambda}}^{n} - \hat{\mathbf{\Lambda}}^{n+1}\right) + \left[\frac{\partial \mathbf{R}^{n}}{\partial \mathbf{Q}^{n}}\right]^T \hat{\mathbf{\Lambda}}^{n} = -\frac{\partial f^{n}}{\partial \mathbf{Q}^{n}}, \quad \text{for } N_m + 1 \le n \le N_{m+1} - 1 \\ \frac{1}{\Delta t}\left(\hat{\mathbf{\Lambda}}^{1} - \hat{\mathbf{\Lambda}}^{2}\right) = -\frac{\partial f^{1}}{\partial \mathbf{Q}^{1}}, \quad \text{for } n = 1, \end{cases} \tag{13}$$

where $\hat{\mathbf{\Lambda}}^n$ and $\tilde{\mathbf{\Lambda}}$ are approximations of the corresponding global-in-time adjoint solutions $\mathbf{\Lambda}^n$ and $\mathbf{\Lambda}^{N_{m+1}+1}$, respectively. Note that the last equation in Eq. (13) is used only on the first subinterval $[T_1, T_2]$ corresponding to $m = 1$. With the local adjoint variables $\hat{\mathbf{\Lambda}}$ satisfying Eq. (13), the local sensitivity derivative on the subinterval $(T_m, T_{m+1}]$ is calculated as follows:

$$\frac{d\hat{L}^m}{d\mathbf{D}} = \begin{cases} \sum\limits_{n=N_m+1}^{N_{m+1}} \frac{\partial f^n}{\partial \mathbf{D}} \Delta t + \sum\limits_{n=N_m+1}^{N_{m+1}} \left[\frac{\partial \mathbf{R}^n}{\partial \mathbf{D}}\right]^T \hat{\mathbf{\Lambda}}^n \Delta t, \quad \text{for } 2 \le m \le M \\ \sum\limits_{n=1}^{N_2} \frac{\partial f^n}{\partial \mathbf{D}} \Delta t + \sum\limits_{n=2}^{N_2} \left[\frac{\partial \mathbf{R}^n}{\partial \mathbf{D}}\right]^T \hat{\mathbf{\Lambda}}^n \Delta t - \left(\frac{\partial \mathbf{Q}^{\text{in}}}{\partial \mathbf{D}}\right)^T \hat{\mathbf{\Lambda}}^1, \quad \text{for } m = 1. \end{cases} \tag{14}$$

The approximate global sensitivity derivative, $\frac{d\hat{L}}{d\mathbf{D}}$, is computed analogously to Eq. (10), i.e.,

$$\frac{d\hat{L}}{d\mathbf{D}} = \sum_{m=1}^{M} \frac{d\hat{L}^m}{d\mathbf{D}}. \tag{15}$$

Once the global sensitivity derivative is available at the last M-th time subinterval, the vector of design variables is updated by using the steepest descent method as follows:

$$\mathbf{D}_{k+1} = \mathbf{D}_k - \delta_k \left(\frac{d\hat{L}}{d\mathbf{D}}\right)_k, \tag{16}$$

where k is the number of an optimization iteration, and δ_k is the optimization step size which is chosen adaptively at each optimization cycle. The steepest descent iterations are repeated until $\|\mathbf{D}_{k+1} - \mathbf{D}_k\| < \epsilon$, where ϵ is a user-specified tolerance and $\|\cdot\|$ is an appropriate norm.

Comparing Eq. (12) with Eq. (13), the following important observation can be made. If $\tilde{\mathbf{\Lambda}}$ in Eq. (13) is set equal to zero, then each local-in-time set of the adjoint equations (13) defined on a given time subinterval is independent of the other adjoint equations defined on the rest of the time interval $[0, T_{\text{final}}]$, thus indicating that the local adjoint equations (13) can be solved sequentially starting from $m = 1$ and marching *forward* one subinterval by another up to $m = M$. Note that the entire set of local adjoint equations on all M time subintervals cannot be solved simultaneously because the adjoint equations require the flow solutions to be available. It should also be emphasized that within each subinterval $(T_m, T_{m+1}]$, the local adjoint equations (13) are integrated backward in time. The local sensitivity derivatives calculated on each subinterval using Eq. (14) are then summed up to give the sensitivity derivative for the entire time interval $[0, T_{\text{final}}]$. Note that the flow adjoint variables obtained with the local adjoint equations (13) for $m = \overline{1, M}$ and the corresponding approximate total sensitivity derivative given by Eq. (15) are not equal to those of the global-in-time formulation given by Eqs. (7, 8), i.e., $\hat{\mathbf{\Lambda}} \neq \mathbf{\Lambda}$ and $d\hat{L}/d\mathbf{D} \neq dL/d\mathbf{D}$ on $(T_m, T_{m+1}]$, where $\mathbf{\Lambda}$ denotes the solution of the global adjoint equations (7). Though the above approach only approximates the original global sensitivity derivative given by Eqs. (7, 8), it reduces the memory cost by a factor of M as compared with the global-in-time formulation. Indeed, since the local adjoint equations on each time subinterval $(T_m, T_{m+1}]$ can be solved independently of the adjoint equations defined on the other subintervals, only the flow solutions for the current subinterval, $\mathbf{Q}^{N_m+1}, \ldots, \mathbf{Q}^{N_{m+1}}$, have to be stored, thus drastically reducing the memory cost.

The second observation resulted from the comparative analysis of Eqs. (7,8, 13, 14) is that the set of the local adjoint equations (13) for $m = \overline{1, M}$ is identical to the global adjoint equations (12) and consequently Eq. (7) if $\tilde{\mathbf{\Lambda}}$ in Eq. (13) is set to be $\mathbf{\Lambda}^{N_{m+1}+1}$. In spite of the fact that this approach provides complete consistency of the local and global adjoint equations, it destroys the locality of the adjoint equations (13) defined on each subinterval and therefore requires the same full storage as the original global-in-time formulation.

The above considerations suggest that $\tilde{\mathbf{\Lambda}}$ in Eq. (13) should be chosen such that it preserves the locality of the local-in-time adjoint equations on each subinterval $(T_m, T_{m+1}]$ and provides a good approximation of $\mathbf{\Lambda}^{N_{m+1}+1}$. To satisfy these constraints, we propose to choose $\tilde{\mathbf{\Lambda}}$ in the following form:

$$\tilde{\mathbf{\Lambda}}_k = \mathbf{\Lambda}_{k-1}^{N_{m+1}+1}, \tag{17}$$

where k is the number of an optimization iteration. In other words, the required vector of adjoint variables at the time level $N_{m+1} + 1$ is taken from the previous iteration of the steepest descent method (16). This choice of $\tilde{\mathbf{\Lambda}}$ significantly reduces the memory cost as compared to the global-in-time adjoint formulation. Indeed, in this case, the flow solution should be stored only at those time levels that belong to the current time subinterval $(T_m, T_{m+1}]$. In addition, M adjoint variables, $\mathbf{\Lambda}_{k-1}^{N_{m+1}+1}$ for $m = \overline{1, M}$, from the previous optimization cycle should also be stored, as follows from Eq. (17). Therefore, the overall memory cost of the proposed methodology is $O(M + N/M)$ flow solutions versus $O(N)$ required for the global-in-time adjoint formulation. This estimate suggests that the optimal value for the number of time subintervals is $M = \sqrt{N}$, where N is the total number of time levels. Another advantage of this method is that if the steepest descent method (16) converges, then the solution of the set of the local-in-time adjoint equations approach to the solution of the original global adjoint equations, thus providing full consistency between the local and global formulations.

The above local-in-time strategy for solving the minimization problem (2, 3) can be formulated in the form of the following algorithm:

ALGORITHM 2.

```
1. Choose $\mathbf{D}_1$, and $M$; set $m = 1$, $k = 1$, $\mathbf{\Lambda}_0^{N_m+1} = 0$ for $m = \overline{1, M}$, and $\frac{d\hat{L}}{d\mathbf{D}} = 0$.
2. Solve Eq. (1) for $\mathbf{Q}_k^{N_m+1}, \ldots, \mathbf{Q}_k^{N_{m+1}}$ forward in time on $(T_m, T_{m+1}]$; store $\mathbf{Q}_k^n$, $n = \overline{N_m + 1, N_{m+1}}$.
3. Solve Eq. (13) with $\tilde{\mathbf{\Lambda}} = \mathbf{\Lambda}_{k-1}^{N_m+1}$ backward in time for $\mathbf{\Lambda}_k^{N_{m+1}}, \ldots, \mathbf{\Lambda}_k^{N_m+1}$; store $\mathbf{\Lambda}_k^{N_m+1}$.
4. Evaluate $\frac{d\hat{L}^m}{d\mathbf{D}}$ using Eq. (14).
5. Set $\frac{d\hat{L}}{d\mathbf{D}} = \frac{d\hat{L}}{d\mathbf{D}} + \frac{d\hat{L}^m}{d\mathbf{D}}$.
6. Set $m = m + 1$, if $m \leq M$ go to step 2; otherwise continue.
7. Calculate $\mathbf{D}_{k+1}$ using Eq. (16).
8. If $\|\mathbf{D}_{k+1} - \mathbf{D}_k\| > \epsilon$ set $m = 1$, $k = k + 1$, $\frac{d\hat{L}}{d\mathbf{D}} = 0$ and go to step 2; otherwise stop.
```

It should be noted that the above local-in-time adjoint-based algorithm can be directly used for solving both time-dependent optimal control problems whose control variables depend on time and design optimization problems whose design variables are independent of time. This is the key difference between the local-in-time method and the receding horizon control technique and its variants (e.g., see Refs. [5, 6]) which are applicable only to optimal control problems, but cannot be used for design optimization.

IV. Numerical Results

We consider simple unsteady 2-D subsonic and supersonic inviscid flows in a channel with a bump to evaluate the performance of the new local-in-time method for design optimization problems. For all test problems considered, the final time, T_{final}, is set to be 1, and the freestream Mach number is given by

$$M(t) = M_0 + \Delta M \cos(\omega t), \tag{18}$$

where M_0 is a mean value of the freestream Mach number, ΔM and ω are an amplitude and frequency of freestream Mach number oscillations. Since the freestream Mach number oscillates in time, the entire flowfield is essentially unsteady. The aerodynamic coefficient in Eq. (3) is the time-dependent pressure coefficient at the lower boundary of the computational domain. The bump shape is described by the following equation:

$$y = d_1\psi_1(x) + d_2\psi_2(x) + d_3\psi_3(x),$$

where $\psi_i(x)$, $i = \overline{1,3}$ are given polynomials satisfying the requirement that the leading and trailing edges of the bump continuously meet the straight lower wall on either side of the bump. Three coefficients d_1, d_2, and d_3 are design variables, i.e., $\mathbf{D} = [d_1, d_2, d_3]^T$.

All test problems are solved on structured grids using a node-centered finite volume code that is first-order accurate both in time and space. At each time step, the nonlinear discrete flow equations are solved by using the Newton's method. For each test, the 2-D Euler equations and their adjoints are converged to machine precision. The local adjoint equations are integrated backward in time and require the solution of the Euler equations to be known only for a current time subinterval which is much smaller that the entire time interval of interest. In the present implementation of the local-in-time method, only the local unsteady solution set on a current time subinterval is held in operating memory, while for the global-in-time optimization method,

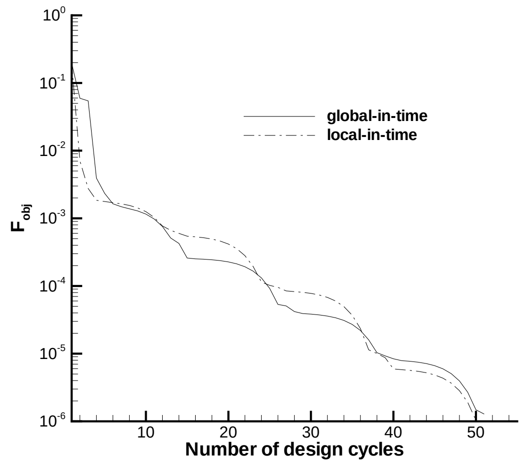

Figure 1. Convergence of the objective functional obtained with the local- and global-in-time methods for the 1st design optimization problem.

the entire unsteady solution history for all time levels has to be stored. The derivatives of $\mathbf{R}$ and f with respect to $\mathbf{Q}$ and $\mathbf{D}$, which are required to form the adjoint equations and the sensitivity derivative, are calculated using the complex variable technique developed by Lyness.[12]

First, we validate the implementation of the local-in-time method and test its performance for the time-dependent design optimization problem (3) in the case that the target flow is feasible, i.e., there is a set

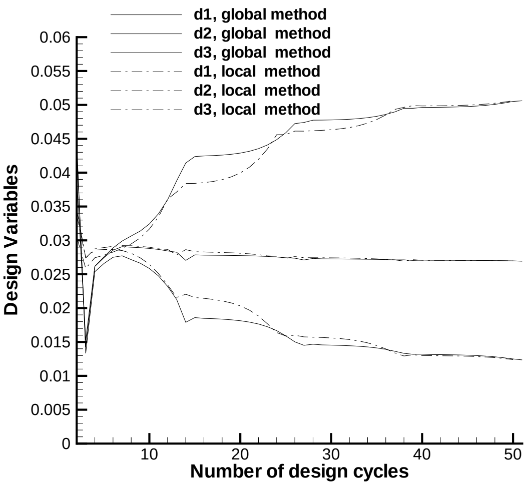

Figure 2. Convergence of the design variables obtained with the local- and global-in-time methods for the 1st design optimization problem.

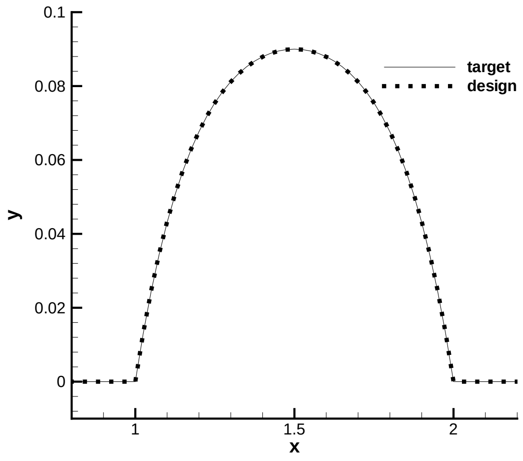

Figure 3. The target and computed bump shapes for the 1st design optimization problem.

of target design parameters that recovers the target flow precisely. Note that in such case, the value of the objective functional at the optimum is zero, and the optimal design variables are expected to be equal to their exact target values. This problem is well suited for evaluation of the performance of optimization methods, because the exact solution is known and the objective functional vanishes at the extremum. The freestream Mach number varies in time according to Eq. (18) with $M_0 = 2$, $\Delta M = 0.2$, and $\omega = 17\pi/9$. Since the flow is supersonic in the entire domain, there are two attached shock waves near the leading and trailing edges of the bump. The target pressure coefficient is obtained by solving the unsteady 2-D Euler equations with the bump parameters/design variables chosen to be $d_1 = 0.05$, $d_2 = 0.03$, and $d_3 = 0.01$. The initial value of each design variable is set to be zero, thus initially, there is no bump on the lower wall. The number of time subintervals, M, used in the local-in-time optimization procedure is set equal to 10, i.e., each interval consists of only one time step. The results presented for this test problem have been obtained using the simplified form of the local-in-time method with $\tilde{\mathbf{\Lambda}} = \mathbf{0}$ in Eq. (13). The optimization is stopped when either the relative change in the value of each design variable becomes smaller than 10^{-4} or the absolute value of the objective functional becomes smaller than 10^{-6}.

The histories of convergence of the objective functional obtained with the local- and global-in-time methods are presented in Fig. 1. Overall, both methods demonstrate a similar convergence rate. The value of

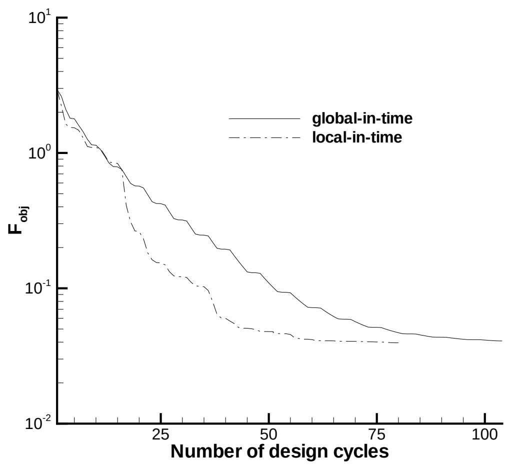

Figure 4. Histories of convergence of the local- and global-in-time adjoint-based optimization methods for the 2nd test problem.

the objective functional monotonically decreases until it becomes less than the specified tolerance. About 50 design cycles are needed to reduce the objective functional by five orders of magnitude. It should be noted that the local-in-time method demonstrates much faster convergence during the first several design cycles than its counterpart. Figure 2 shows convergence histories of all three design variables during optimization. The most important conclusion that can be drawn from this comparison is that the local- and global-in-time methods converge to the same solution. From this standpoint, the solution obtained with the local-in-time method is optimal with respect to the original optimization problem (2). It should also be noted that all the design variables approach to their target values. In spite of the fact that the second bump parameter, d_2, is slightly overpredicted while the third one, d_3, is underpredicted, the optimal shape is practically indistinguishable from the target bump, as one can see in Fig. 3. From the comparisons presented above it follows that the locally optimal method does not only converge to the same optimal solution computed using the global-in-time method, but also reduces the memory cost by a factor of 10 as compared with its conventional counterpart.

The second test problem has been chosen so that the target flow is infeasible. In other words, the value of the objective functional at the optimum is not equal to zero. For this test case, the target pressure coefficient on the lower wall is set to be $C_P^{\text{target}}(t) = 0.3\sin(0.5\pi t)$. The freestream Mach number is given by Eq. (18) with $M_0 = 0.5$, $\Delta M = 0.1$, and $\omega = 17\pi/9$. As a result, the entire flowfield is subsonic for all time levels. Similar to the previous test case, $\tilde{\mathbf{\Lambda}}$ in Eq. (13) is set to be zero at each design cycle. For both the local- and global-in-time optimization procedures, the initial values of the three design parameters are chosen to be zero and the convergence tolerance for each design variable is set equal to 10^{-4}.

Figure 4 shows the convergence histories of the local- and global-in-time optimization techniques. Each optimization method has reduced the value of the objective functional by almost two orders of magnitude during the design. However, the global-in-time method requires about as twice as many optimization cycles to reach the minimum as compared with the local-in-time optimization procedure. Furthermore, the memory cost of the local-in-time adjoint-based optimization method is 10 times cheaper than that of its conventional counterpart. This is because on each time subinterval of the local-in-time method, the local adjoint equations are solved over one time step, while for the global-in-time method, the flow and adjoint equations are integrated over all 10 time levels. Note that for practical applications that require $10^3 - 10^4$ time steps to resolve the unsteady flow dynamics, the local-in-time method can provide three to four orders of magnitude reduction in the memory cost as compared with the global-in-time adjoint-based method. Figure 5 compares the convergence histories of the three design parameters. As in the previous test case, the local- and global-in-time adjoint-based methods converge to the same optimal solution. To compare the optimal solutions obtained with the local- and global-in-time optimization techniques and to illustrate how they converge to the target solution, the computed, target, and initial lift coefficients are presented in Fig. 6. The difference between the initial lift coefficient and its target value is of the order of $O(1)$, thus indicating that the flowfield

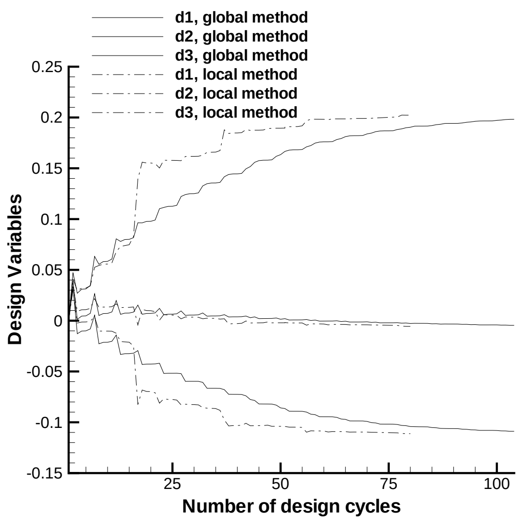

Figure 5. Convergence of the design variables obtained with the local- and global-in-time methods for the 2nd test problem.

changes significantly during the design. The optimal solutions obtained with the local- and global-in-time methods are almost indistinguishable from each other. Another interesting observation is that in spite of the fact that the target flow is infeasible, the lift coefficients computed with both optimization techniques agree very well with the target lift coefficient over the entire time interval considered.

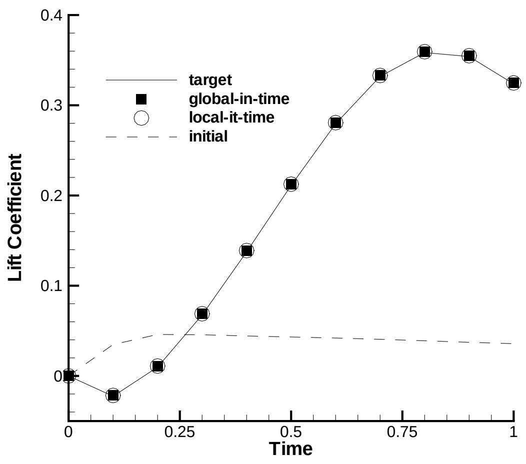

Figure 6. Comparison of the optimal lift coefficients computed using the local and global-in-time methods with their initial and target values for the 2nd test problem.

V. Conclusion

The new local-in-time adjoint-based method for optimization of unsteady compressible flows has been developed. In contrast to conventional gradient methods based on the adjoint formulation that requires the solution of the flow equations to be available for all time steps, the local-in-time optimization algorithm solves the local adjoint equations sequentially on each time subinterval to form the global sensitivity derivative. Since each set of the local adjoint equations is integrated backward in time over only a small time subinterval, its memory cost is M times smaller than that of the global-in-time adjoint formulation, where M is the number of time subintervals used. In the limit, each time subinterval can consist of a single time step, thus reducing the memory cost to the level equivalent to that of the steady state adjoint formulation. Another very attractive feature of the new local-in-time optimization algorithm is that it converges to the same optimal solution obtained with the global-in-time adjoint-based method, as has been shown in our numerical experiments. For all test problems considered, the local-in-time optimization method provides at least the same convergence rate as compared with its conventional counterpart. These properties of the local-in-time adjoint-based methodology indicate that it can be used for solving a broad spectrum of realistic large-scale optimal control and design optimization problems arising in various unsteady aerodynamic applications.

References

[1]Anderson W. K. and Bonhaus D. L., "An implicit upwind algorithm for computing turbulent flows on unstructured grids," *Computers and Fluids*, Vol. 23 pp. 1-21, 1994.

[2]Roe, P. L., "Approximate Riemann Solvers, Parameter Vectors, and Differential Schemes," *J. of Comput. Phys.,* Vol. 43, No. 2, pp. 357-372, 1981.

[3]Nielsen, E. J., Diskin B., and Yamaleev, N. K. "Discrete Adjoint-Based Design Optimization of Unsteady Turbulent Flows on Dynamic Unstructured Grids," submitted to 19th AIAA Computational Fluid Dynamics Conference, San Antonio, TX, 22-25 June, 2009.

[4]Yamaleev, N. K., Diskin B., and Nielsen, E. J. "Adjoint-Based Methodology for Time-Dependent Optimization," *AIAA Paper* 2008-5857, 2008.

[5]Hou, L. S., and Yan, Y., "Dynamics and Approximations of a Velocity Tracking Problem for the Navier-Stokes Flows with Piecewise Distributed Controls," *SIAM J. Control Optim.*, Vol. 35, No. 6, pp. 1847–1885, 1997.

[6]Hinze, M., and Kunisch, K., "On Suboptimal Control Strategies for the Navier-Stokes Equations," in *Control and Partial Differential Equations*, ESAIM, Paris, pp. 181–198, 1998.

[7]Nadarajah, S. K., Jameson, A., "Optimal Control of Unsteady Flows Using a Time Accurate Method," *AIAA Paper* 2002-5436, 2002.

[8]Collis, S. S., Ghayour, K., Heinkenschloss, M., Ulbrich, M., Ulbrich, S., "Optimal Control of Unsteady Compressible Viscous Flows," *Int. J. Numer. Meth. Fluids*, Vol. 40, No. 11, pp. 1401-1429, 2002.

[9]Joslin, R.D, Gunzburger, M.D., Nicolaides, R.A., Erlebacher, G., and Hussaini, M.Y., "Self-Contained Automated Methodology for Optimal Control," *AIAA J.* Vol. 35, No. 5, pp. 816–824, 1997.

[10]Mani, K., Mavriplis, D. J., "An Unsteady Discrete Adjoint Formulation for Two-Dimensional Flow Problems with Deforming Meshes," *AIAA Paper* 2007-60, 2007.

[11]Rumpfkeil, M.P., and Zingg, D.W., "A General Framework for the Optimal Control of Unsteady Flows with Applications," *AIAA paper*, 2007-1128, 2007.

[12]Lyness, J. N. and Moler, C. B., "Numerical Differentiation of Analytic Functions," *SIAM J. Numerical Analysis*, Vol. 4, pp. 202-210, 1967. 1967, pp. 124–134.