Challenges in Adjoint-Based Aerodynamic Design for Unsteady Flows

Eric J. Nielsen
FUN3D Development Team
NASA Langley Research Center
USA

http://fun3d.larc.nasa.gov
It’s A Multidisciplinary World

Ablation  
Acoustics  
Aerodynamics  
Flight Dynamics  
Materials  
Optics  
Performance  
Propulsion  
Radiation  
Structures

Mars InSight Lander  
Gulfstream G550  
Ares 1-X  
NASA/Boeing Truss-Braced Wing  
Inflatable Decelerators
Today’s design approaches typically rely on Euler and RANS simulations, each requiring $O(10^2)$-$O(10^3)$ CPU hours on moderate HPC resources.

Current projection is full aircraft LES as a grand challenge problem in the 2045 timeframe using an entire leadership-class machine, with DNS following in 2080*

Hybrid RANS-LES simulations are the current state of the art and may require $O(10^7)$ CPU hours on large HPC resources.

Accurate predictions for many aerospace concepts require at least hybrid RANS-LES:

**Pushing these methods into the design cycle is critical**

Active Flow Control
Where should control jets be located?
At what orientation?
What should the unsteady blowing profiles look like?
What is the optimal phase difference between jets?
How should the outer mold line be altered?

Goal: Enable formal, physics-based design optimization based on large-scale computational simulations of vehicles where we may have no a priori knowledge nor experience

…and what should my grid look like?
• Systematic design of a complete vehicle may involve thousands of design variables

• The number of function evaluations required by zeroth-order (e.g., sampling) optimization techniques increases dramatically with only a few design variables

→ Gradient-based methods are the only feasible approach

**Bear in mind that we have not even touched on:**
• Robust design optimization
• Multidisciplinary optimization
• Uncertainty quantification
• …

*We are only at the tip of the iceberg!*
Conventional sensitivity analysis techniques such as finite differencing or direct differentiation consider a perturbation to a single input parameter. This effect is then propagated through the simulation to ultimately determine a single element of the desired gradient vector. This class of methods is referred to as *forward-mode differentiation*. These methods can effectively provide sensitivities of many outputs with respect to a single input. However, the cost of these approaches scales linearly with the number of design variables. E.g., for a problem with 1,000 variables, central differencing will require 2,000 (very accurate!) simulations just to obtain a single gradient vector. These approaches are prohibitively expensive in our context. So how can we efficiently compute sensitivity information for thousands of simulation parameters?
Motivation for Adjoint Methods

- The adjoint approach flips the entire sensitivity analysis upside down by solving an auxiliary PDE and instead pushing the dependence on the number of design variables to the very end of the process.
- In this manner, everything is done backwards; hence, adjoint methods are often referred to as reverse-mode differentiation.

Adjoint methods can provide sensitivities of an output function for virtually unlimited numbers of input parameters at the cost of a single additional simulation.
Adjoint Solution Example

F-15 Configuration

- Transonic turbulent flow over modified F-15 configuration
- Propulsion effects included as well as simulated aeroelastic deformations of canard/wing/h-tail
- Objective is lift-to-drag ratio
- Adjoint solution indicates where objective is sensitive to perturbations in both space and time
Adjoint Solution Example
Wind Turbine Configuration

- Incompressible turbulent flow over NREL Phase VI wind turbine
- Overset grids used to model rotating blade system
- Objective function is based on the torque

Forward Solution
Reverse Solution
Some Challenges
The Unsteady Adjoint Equations

Complexity
- Considerably more involved than the Navier-Stokes equations
- Every line of the baseline code must be differentiated with respect to flow solution, grid coordinates, and design variables
- Tremendous amount of software infrastructure required
- Implemented by hand and verified using complex variables

Sheer Expense
- Full linearizations must be evaluated at every time step

Page 1 of 4 of the adjoint equations derived and implemented in:

Since the adjoint equations must be integrated backwards in time, we must have the forward solution available at every time plane.

**Possible Approaches**

- **Brute force**: Store the entire forward solution.
- **Recompute**: Store the forward solution periodically and recompute intermediate time steps as needed.
- **Approximate**: Store the forward solution periodically and interpolate intermediate time planes somehow.
In FUN3D, we store all of the forward data to disk

The amount of data adds up fast – consider a small example:
- 50,000,000 grid points and 10,000 physical time steps
- Assume a 1-equation turbulence model (5+1 unknowns per grid point)
- Dynamic grids (3 additional unknowns per grid point)
  \[ \rightarrow 50,000,000 \times 10,000 \times (6+3) \times 8 \text{ bytes} = 36 \text{ Terabytes} \]

This amount of data is not prohibitively large, but we need to run much bigger problems, say \(10^9\) grid points with \(10^6\) time steps.

So far, the challenge has been efficiently getting the data to/from the disk at every time step.
• Approaches used to write conventional checkpoint files are prohibitively expensive
• FUN3D uses parallel, asynchronous, direct access read/writes from every rank
  – Flow solver is writing the previous time plane while the current time step is computing
  – Adjoint solver is pre-fetching earlier time planes while the current time step is computing
• This strategy has performed well so far, but is not infinitely scalable
Application Examples
• Such simulations are tremendously complex; here we are only doing aero
• Overset grid system consists of 9,262,941 nodes / 54,642,499 tetrahedra
• Compressible RANS: \( M_{\text{tip}}=0.64 \), \( Re_{\text{tip}}=7.3\text{M} \), \( \mu=0.37 \), \( \alpha=0.0^\circ \)
• Blade pitch has child motion governed by pilot collective and cyclic controls:

\[ \theta = \theta_c + \theta_{1c} \cos \psi + \theta_{1s} \sin \psi \]
Example: UH-60A Blackhawk

**Problem Definition and Results**

- **Objective** is to maximize time-averaged lift while satisfying trim constraints:
  
  $$\min f = \left[ \left( \frac{1}{360} \sum_{n=361}^{720} C_L^n \right) - 2.0 \right]^2 \Delta t$$  

  such that

  $$g_1 = \frac{1}{360} \sum_{n=361}^{720} C_{M_x}^n \Delta t = 0$$

  $$g_2 = \frac{1}{360} \sum_{n=361}^{720} C_{M_y}^n \Delta t = 0$$

- Separate adjoint solutions required for all three functions
- 67 design variables include 64 thickness and camber variables across the blade planform, plus collective and cyclic control inputs

![Graph showing objective function and moment constraints vs. design cycle.](image)

- Feasible region is quickly located
- Both moment constraints are satisfied within tolerance at the optimal solution
- Final controls: $\theta_c = 6.71^\circ$, $\theta_1 = 2.58^\circ$, $\theta_s = -7.00^\circ$

<table>
<thead>
<tr>
<th></th>
<th>(\bar{C}_L)</th>
<th>Flow Solves (2 hrs)</th>
<th>Adjoint Solves (3 hrs)</th>
<th>Total Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline</td>
<td>0.023</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Design</td>
<td>0.103</td>
<td>4</td>
<td>4</td>
<td>0.8 days (38,400 CPU hrs)</td>
</tr>
</tbody>
</table>
**Example: UH-60A Blackhawk**

**Results**

Lift has gone up significantly; vehicle is trimmed in both pitch and roll.

[Graphs showing lift and pitching moment over time with labeled intervals for cost and constraint functions.]
Multidisciplinary adjoint has been very successful for sonic boom mitigation – discrete sensitivities of ground-based metrics to aircraft geometry

- Recently extended to include atmospheric UQ
- Many other disciplines being considered / pursued

**Sonic Boom Mitigation**

![Diagram](image)

- **FUN3D Adjoint Solver**
- **FUN3D Flow Solver**
- **CFD off-body $dp/p$**
- **sBOOM**
  - (Augmented Burgers Equation for Propagation)
- **sBOOMAdjoint**
- **Ground Signature, Loudness**
- **Sensitivity Analysis**
- **Baseline**
- **Optimal**

**Loudness at ground level reduced from 65.2 to 59.8 dBA**

$$\frac{\partial (\text{loudness})}{\partial \hat{n}}$$
Mesh Adaptation Examples

High-Lift

Sonic Boom

Propulsion

Reentry

Courtesy Chris Heath
A Remaining Challenge: Chaos

**Wish to compute sensitivities of infinite time averages for chaotic flows (Hybrid RANS-LES, LES, DNS)**

- Theory states these sensitivities are well-defined and bounded

**Why does conventional approach not work?**

For chaotic flows,
- The finite time average approaches the infinite time average
- The sensitivity for a finite time average does not approach the sensitivity for the infinite time average

*Chaotic shedding for 0012*

\[ M_\infty = 0.1 \quad Re = 10,000 \quad \alpha = 20^\circ \]

*Adjoint solution grows exponentially in reverse time*

\[
\| \Delta \|_2
\]
• Least-Squares Shadowing (LSS) method proposed by Wang and Blonigan (MIT)
  • Key assumption is ergodicity of the simulation: long time averages are essentially independent of the initial conditions
  • Also assumes existence of a shadowing trajectory
• The LSS formulation involves a linearly-constrained least squares optimization problem which results in a set of optimality equations
• The LSS adjoint equations are a globally coupled system in space-time
• To date, work at MIT has focused on solutions of this system for academic dynamical systems containing $O(1)$ state variables
• Langley and MIT are collaborating to explore the extension to CFD systems: enormous computational challenge for even the smallest of problems
A Remaining Challenge: Chaos
Least-Squares Shadowing (MIT)

- Shedding NACA 0012
  $M_{\infty}=0.1$  $Re=10,000$  $\alpha=20^\circ$
  102,940 grid points

- Goal is to compute an AOA sensitivity that would allow us to maximize the time-averaged lift over final 1,000 time steps

**Instantaneous Lift vs Time**

**Finite Time Average of Lift vs Alpha**
• FUN3D used to output data for use in LSS solver
  • Nonlinear residual vectors; Jacobians of residual, objective function
  • For this tiny problem, this is 1.1 TB of raw data

• Dimension of the resulting LSS matrix problem:
  102,940 grid points x 5 DOFs
  x 2,000 time planes = 1.03 billion

• Stand-alone LSS solver has been developed where decomposition is performed in time with a single time plane per core

• Global GMRES solver used with a local ILU(0) preconditioner for each time plane – has proven vastly inadequate

• Required ~10 hours on 2,000 cores

This is a toy problem – target simulations are $10^6$ larger!
Desired matrix dimension = $10^9 \times 10^6 = 10^{15}$
“If you build it, we will come…”

Thank you to the organizers for having me!