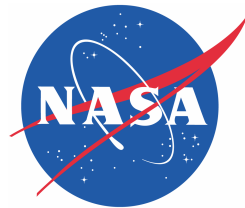


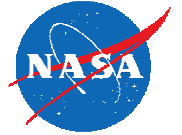
Integrated Design of an Active Flow Control System Using a Time-Dependent Adjoint Method



Eric J. Nielsen and William T. Jones
Computational AeroSciences Branch

<http://fun3d.larc.nasa.gov>

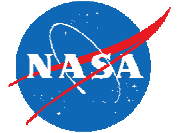
Introduction



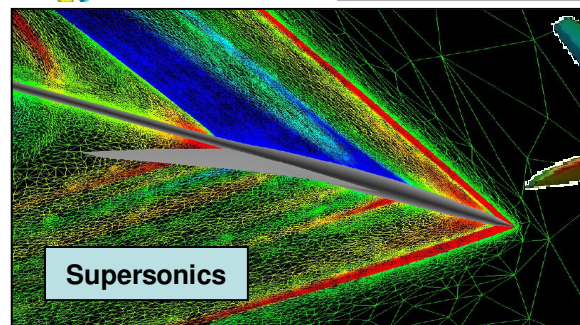
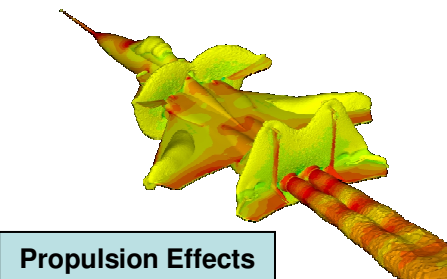
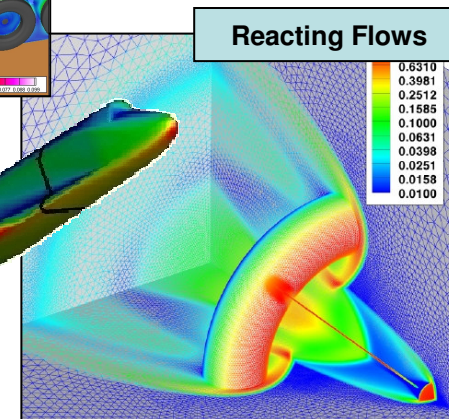
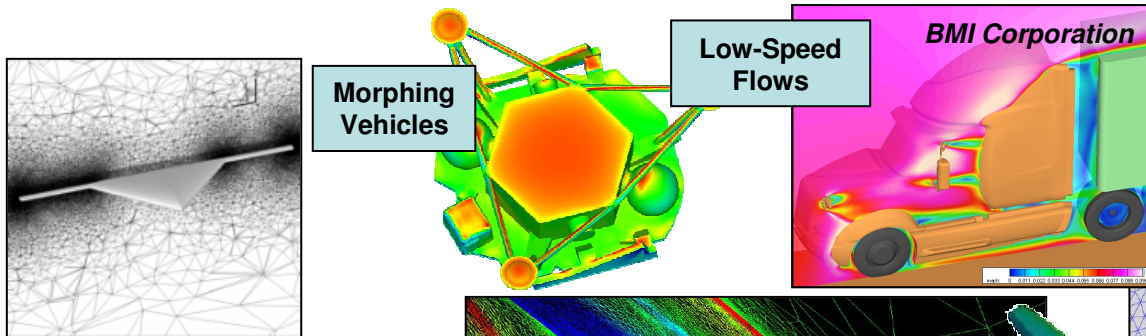
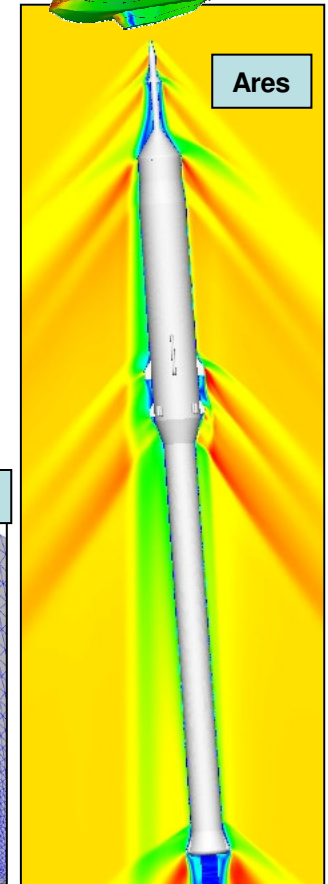
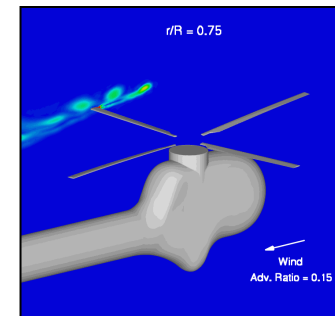
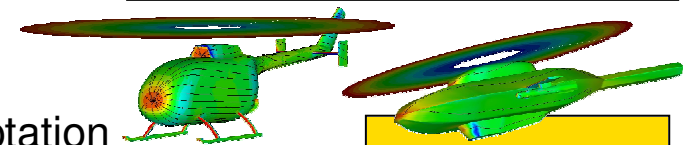
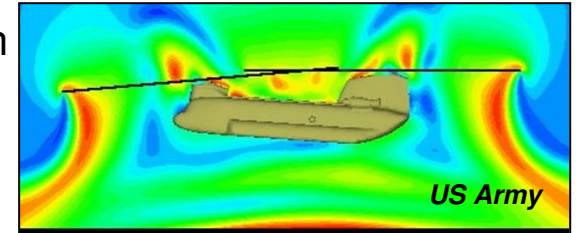
- Adjoint methods provide rigorous sensitivity analyses for CFD simulations at the nominal cost of a single flow solution
- Use of adjoint methods for design of steady aerodynamic problems has exploded over the past two decades
 - FUN3D team has worked extensively in this area for 15 years and are at the forefront of the technology/implementation
- Adjoint methods for general unsteady flows still in their infancy
 - Practical limitations are the biggest impediments
 - **Entire** unsteady solution must be available – not just last few back planes
 - The cost of a single unsteady simulation can be considerable
 - Despite algorithmic efficiencies, the cost of formal adjoint-based design optimization for general unsteady flows can still be **tremendous**
- Formal adjoint-based design would seem to be a natural fit for AFC devices
 - AFC devices being explored for wide variety of uses
 - Often rely on heuristic methods to choose device settings and placement
 - Often done as an afterthought on an existing configuration
 - Formal physics-based optimization of the coupled system may yield unforeseen performance benefits

FUN3D Core Capabilities

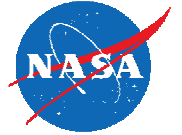
<http://fun3d.larc.nasa.gov>



- Solves 2D/3D steady and unsteady Euler and RANS equations on node-based mixed element grids for compressible and incompressible flows
- Supports numerous internal/external efforts across speed range
- General dynamic mesh capability: any combination of rigid/overset/morphing grids, including 6-DOF effects
- Aeroelastic modeling w/ mode shapes, full FEM, CC, etc
- Constrained/multipoint adjoint-based design and mesh adaptation
- Modern software practices including 24/7 testing
- Linear scaling through thousands of cores
- Capabilities fully integrated, very responsive support team, online documentation, training videos, tutorials, etc



Governing Equations



Flow Equations

- Unsteady RANS equations with BDF2_{opt} in time:

$$V^n \frac{a\mathbf{Q}^n + b\mathbf{Q}^{n-1} + c\mathbf{Q}^{n-2} + d\mathbf{Q}^{n-3}}{\Delta t} + \mathbf{R}^n = 0 \quad \mathbf{R} \equiv \oint_{\partial V} (\mathbf{F}_i - \mathbf{F}_v) \cdot \hat{\mathbf{n}} dS$$

- BDF2 and BDF1 used at $n=2$, $n=1$; initial solution specified at $n=0$

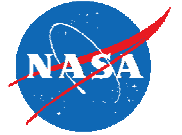
Grid Equations

- FUN3D's adjoint capability can handle dynamic meshes; however, mesh is static for this work and deforms only as a result of the optimization procedure
- Modeled as an elastic medium which obeys the elasticity relations of solid mechanics
- Auxiliary PDE solved to determine mesh coordinates after each shape update:

$$\mathbf{G}(\mathbf{X}, \mathbf{D}) = \mathbf{X}_{surf} - \mathbf{K}\mathbf{X}$$

- $\mathbf{K}$ is the elasticity coefficient matrix and is based on the local cell geometry and proximity to the surface

Objective/Constraint Functions

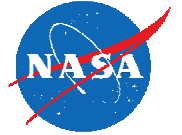


$$f_i = \sum_{n=N_i^1}^{N_i^2} f_i^n \Delta t, \text{ where } f_i^n = \sum_{j=1}^{J_i} \omega_j (C_j - C_j^*)^{p_j}$$

ω = weight C = aero coeff
 p = power C^* = target aero coeff

- User specifies time interval for summation
- User may specify which boundary patch in the grid (or all) to which each function applies
- Constraints may be explicit or added as “penalties”
- Multipoint/multiobjective: as many composite functions/constraints as desired
 - Only limited by particular optimization package (couple hours to hook up COTS optimizer)
 - Adjoint for multiple functions/constraints computed simultaneously

Unsteady Adjoint Equations for Static Grids



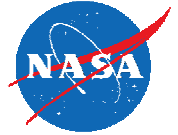
As in steady cases, construct Lagrangian using an objective function and adjoint variables multiplying the governing equations as constraints:

$$\begin{array}{ll} \Lambda_f \equiv \text{Flowfield adjoint variable} & \mathbf{R} \equiv \text{Spatial residual} \\ \Lambda_g \equiv \text{Grid adjoint variable} & f \equiv \text{Objective function} \end{array}$$

$$\begin{aligned} L(\mathbf{D}, \mathbf{Q}, \mathbf{X}, \Lambda_f, \Lambda_g) = & \underbrace{\sum_{n=1}^N f^n \Delta t}_{\text{Objective Function}} + \sum_{n=1}^N [\Lambda_f^n]^T \underbrace{\left(V^n \frac{a\mathbf{Q}^n + b\mathbf{Q}^{n-1} + c\mathbf{Q}^{n-2} + d\mathbf{Q}^{n-3}}{\Delta t} + \mathbf{R}^n \right)}_{\text{Flow Equations}} \Delta t \\ & + \underbrace{\left(f^0 + [\Lambda_f^0]^T \mathbf{R}^{in} \right) \Delta t}_{\text{Similar terms for initial conditions}} + [\Lambda_g]^T \underbrace{\mathbf{G} \Delta t}_{\text{Grid Equations}} \end{aligned}$$

- Linearize with respect to $\mathbf{D}$, isolate the coefficients of $\partial \mathbf{Q}^n / \partial \mathbf{D}$ and $\partial \mathbf{X} / \partial \mathbf{D}$, and equate them to zero

Unsteady Adjoint Equations for Static Grids



**Flowfield
Adjoint
Equations**

$$\frac{V}{\Delta t}(a\Lambda_f^n + b\Lambda_f^{n+1} + c\Lambda_f^{n+2} + d\Lambda_f^{n+3}) + \left[\frac{\partial \mathbf{R}^n}{\partial \mathbf{Q}^n} \right]^T \Lambda_f^n = - \left[\frac{\partial f^n}{\partial \mathbf{Q}^n} \right]^T \quad 3 \leq n \leq N$$

Similar expressions for $n=2,1,0$

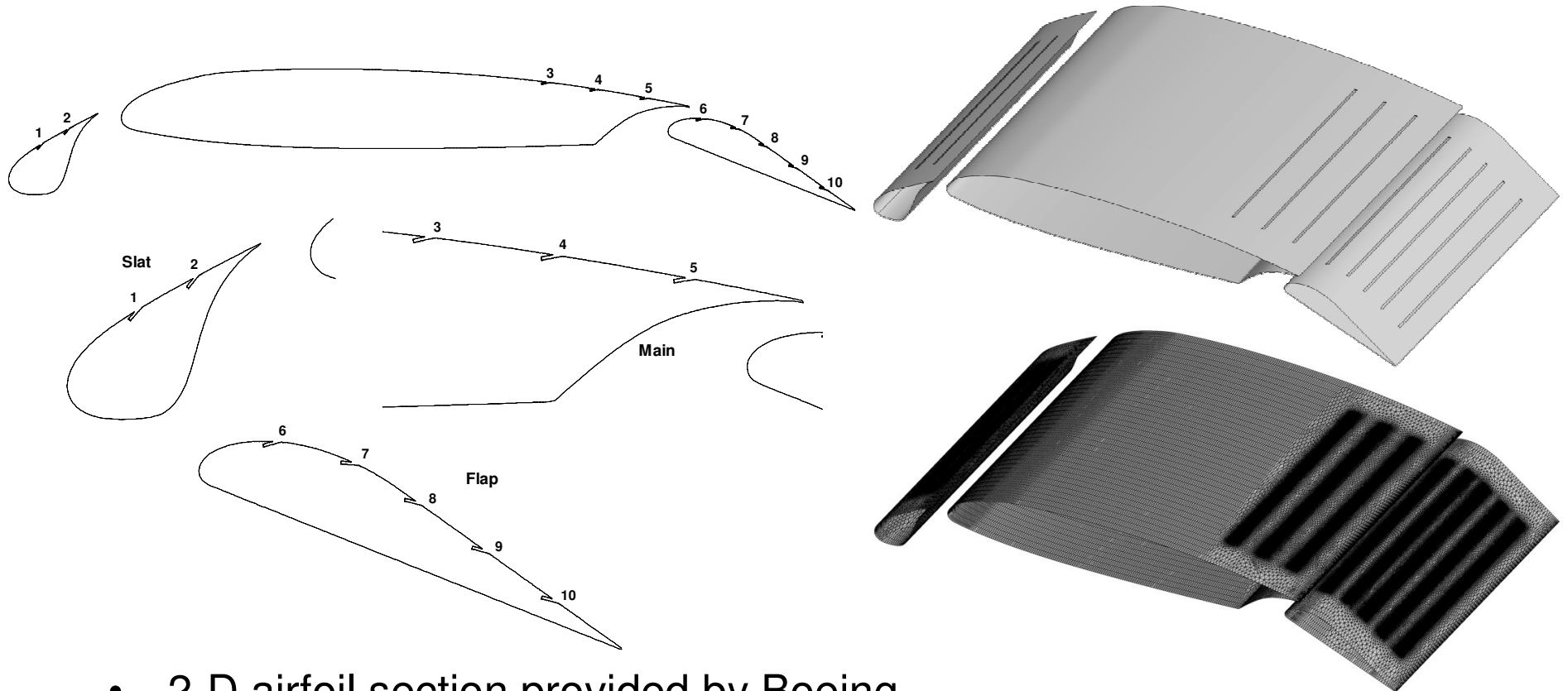
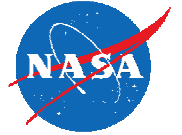
Grid Adjoint Equation

$$\begin{aligned} \left[\frac{\partial \mathbf{G}}{\partial \mathbf{X}} \right]^T \Lambda_g = & \sum_{n=3}^N \left\{ \left[\frac{\partial f^n}{\partial \mathbf{X}} \right]^T + \left[\frac{\partial V}{\partial \mathbf{X}} \frac{a\mathbf{Q}^n + b\mathbf{Q}^{n-1} + c\mathbf{Q}^{n-2} + d\mathbf{Q}^{n-3}}{\Delta t} + \frac{\partial \mathbf{R}^n}{\partial \mathbf{X}} \right]^T \Lambda_f^n \right\} + \left[\frac{\partial f^2}{\partial \mathbf{X}} \right]^T \\ & + \left[\frac{1}{2} \frac{\partial V}{\partial \mathbf{X}} \frac{3\mathbf{Q}^2 - 4\mathbf{Q}^1 + \mathbf{Q}^0}{\Delta t} + \frac{\partial \mathbf{R}^2}{\partial \mathbf{X}} \right]^T \Lambda_f^2 + \left[\frac{\partial f^1}{\partial \mathbf{X}} \right]^T + \left[\frac{\partial V}{\partial \mathbf{X}} \frac{\mathbf{Q}^1 - \mathbf{Q}^2}{\Delta t} + \frac{\partial \mathbf{R}^1}{\partial \mathbf{X}} \right]^T \Lambda_f^1 + \left[\frac{\partial f^0}{\partial \mathbf{X}} \right]^T + \left[\frac{\partial \mathbf{R}^{in}}{\partial \mathbf{X}} \right]^T \Lambda_f^0 \end{aligned}$$

**Sensitivity
Equation**
$$\frac{dL}{d\mathbf{D}} = \sum_{n=1}^N \left(\frac{\partial f^n}{\partial \mathbf{D}} + \left[\Lambda_f^n \right]^T \frac{\partial \mathbf{R}^n}{\partial \mathbf{D}} \right) \Delta t + \left(\frac{\partial f^0}{\partial \mathbf{D}} + \left[\Lambda_f^0 \right]^T \frac{\partial \mathbf{R}^{in}}{\partial \mathbf{D}} + \left[\Lambda_g \right]^T \frac{\partial \mathbf{G}}{\partial \mathbf{D}} \right) \Delta t$$

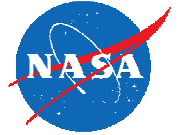
- Dependencies on all time levels in stencil must be linearized
- Traditional steady terms shown in red
- Optional source terms not explicitly shown; can be included in $\mathbf{R}$
 - Actuator disks, noninertial terms, chemistry, etc.

Test Case



- 2-D airfoil section provided by Boeing
- 10 jet orifices: $0.002c$ in width, oriented 20° to local tangent
- Extruded one chord in spanwise direction at zero sweep angle
 - Jet slots are $0.75c$ wide, centered spanwise
- Grid generated using GridEx and VGRID
 - Consists of 8,823,297 nodes / 50,870,813 tetrahedra

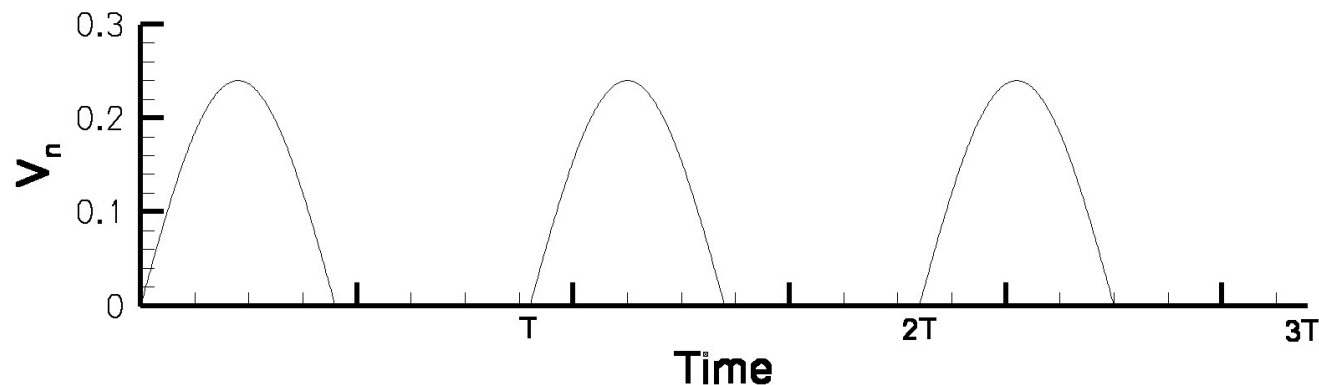
Test Case



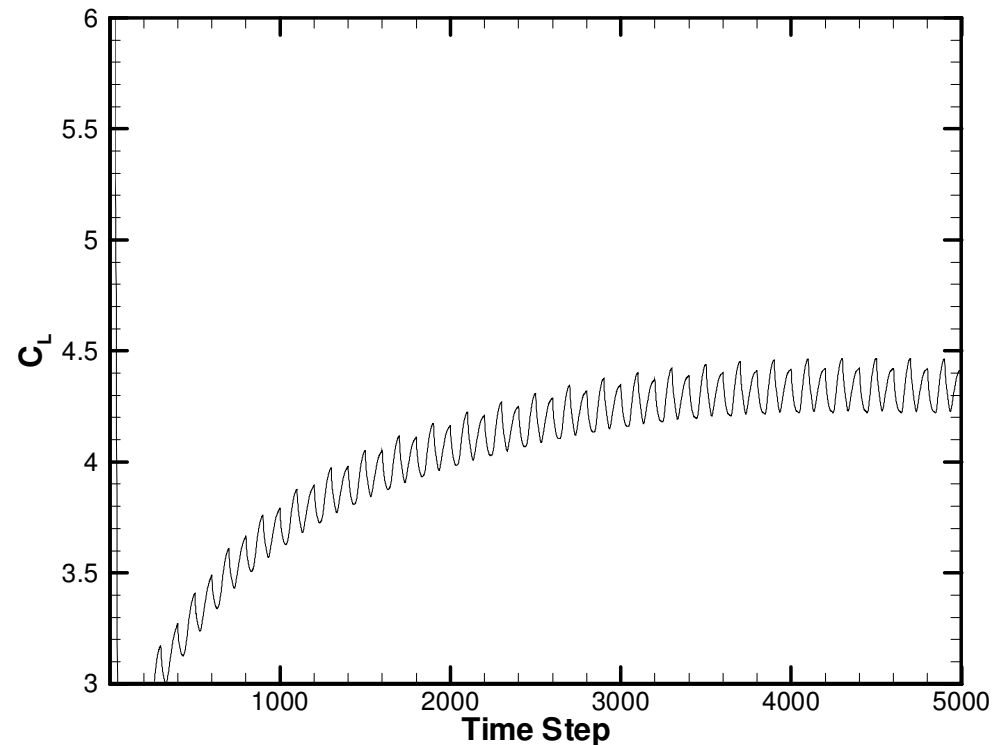
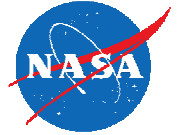
- $M_\infty=0.1$, $\alpha=19^\circ$, $Re_c=3 \times 10^6$
- Rather than model actual piston diaphragm motion, jet actuation is achieved with an oscillatory blowing condition:

$$V_n = \max[0, \bar{V} \sin(\omega t + \varphi)], \quad \bar{V} = 0.24$$

- $F^+=1.52$ (20 Hz), each adjacent jet is 180° out of phase
- BDF2_{opt} with 10 subiterations
- Lift does not change significantly with more than 100 steps per blowing cycle; however, Δt is chosen to correspond to 200 steps per cycle to resolve higher blowing frequencies (more later)

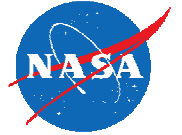


Baseline Aerodynamic Performance



- 5,000 physical time steps (25 jet blowing cycles) required to reach periodic behavior from freestream initialization
- Mean value of lift coefficient is 4.33 over final 1,000 time steps
- Clean wing lift coefficient is 3.79

Design Variables: Blowing Parameters

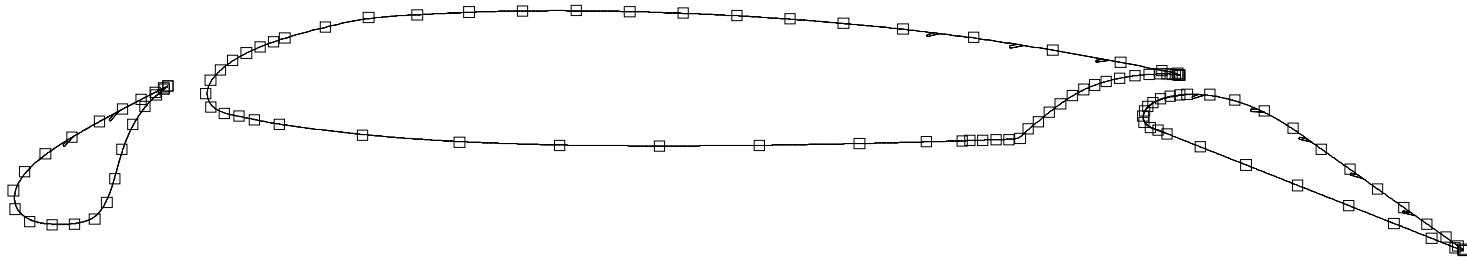
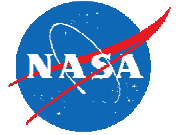


$$V_n = \max[0, \bar{V} \sin(\omega t + \varphi)]$$

- For each of the ten jets, the frequency, velocity, and phase of the blowing actuation are available design variables subject to the following bounds:

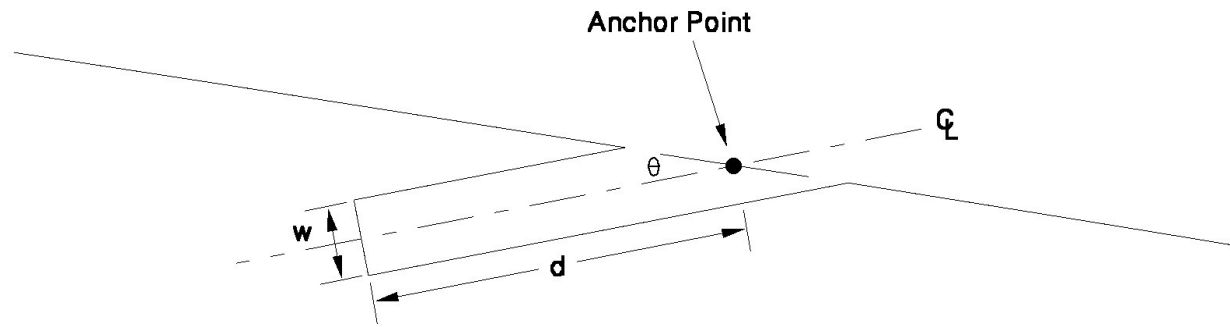
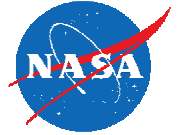
$$0.76 \leq F^+ \leq 3.04 \quad 0.0 \leq \bar{V} \leq 0.48 \quad 0^\circ \leq \varphi \leq 180^\circ$$

Design Variables: Shape Parameterization



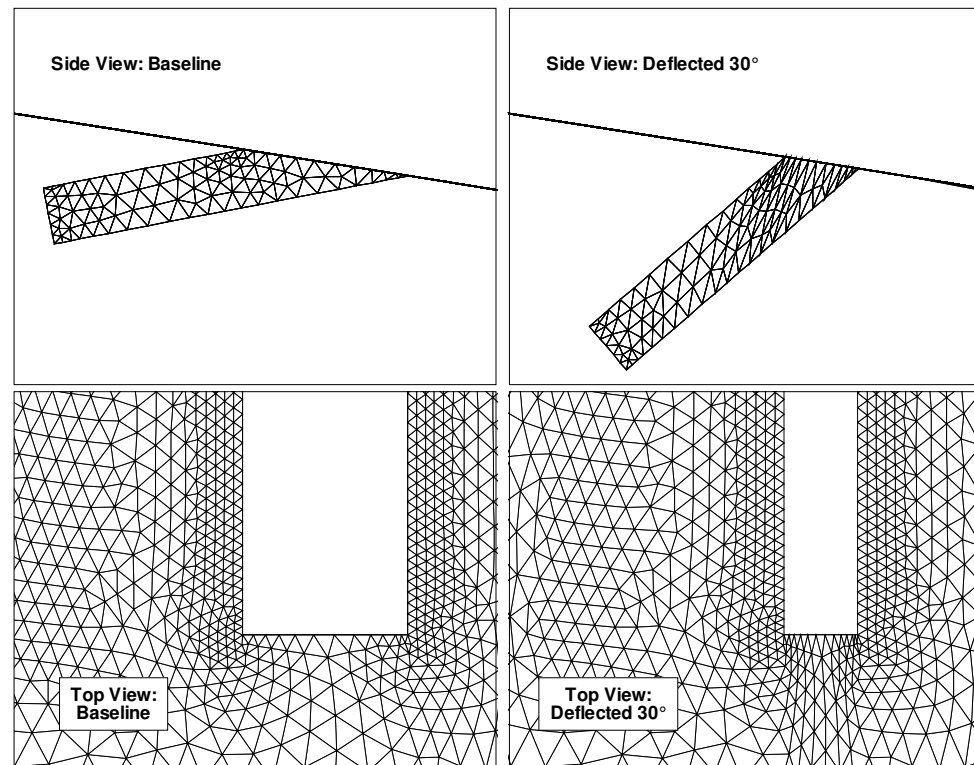
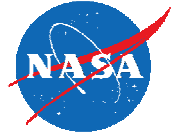
- Specialized pseudo-2D geometry tool developed for this application
 - not intended for general use
- More general packages such as Massoud could be extended to handle features sliding along morphing surfaces, but this is not done here
- Based on the use of B-splines with an arc length parameterization
- The displacements of the B-spline control points normal to the surface are used as design variables to control the wing shape
 - Multiple B-splines can be fit to the data to achieve good fits while minimizing the number of control points
 - Individual B-splines then concatenated to form a single composite curve for the body
- Body translation and rotation may also be used as design variables

Design Variables: Jet Parameterization



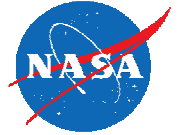
- An anchor point is established for each jet
- The parametric location of this anchor point is available as a design variable in order to reposition the jets along the chord
- The incidence angle θ of each jet is also a design variable
- Jet width and depth are fixed

Surface Grid Deformation

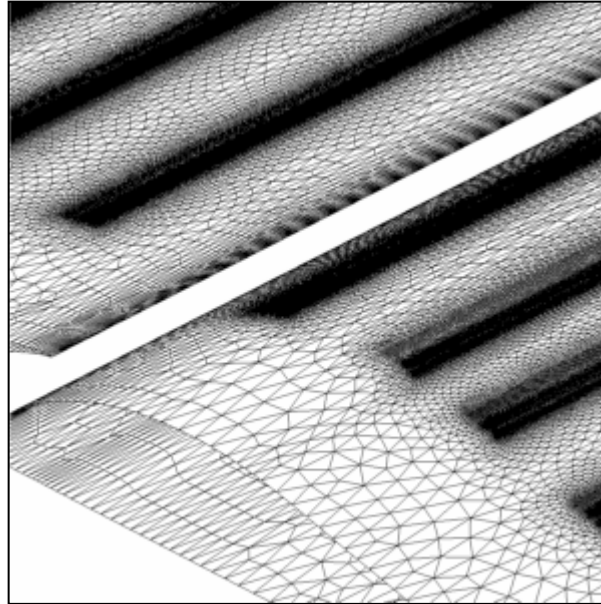
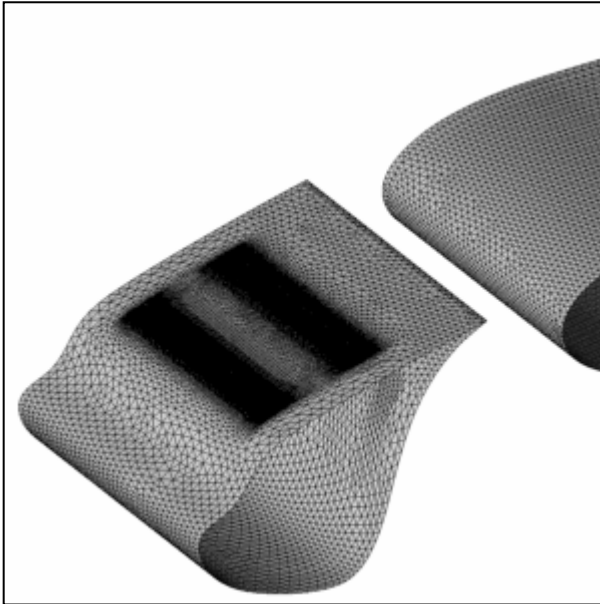


- As an anchor point is repositioned, the jet is moved along with it at the specified incidence angle
- Jet sidewalls are extended/reduced as needed to maintain depth
 - Points along sidewall edges are redistributed linearly and points interior to the sidewall patches are smoothed
- Points on exterior wing surface smoothed along chord as a final step

Summary of Available Shape Changes

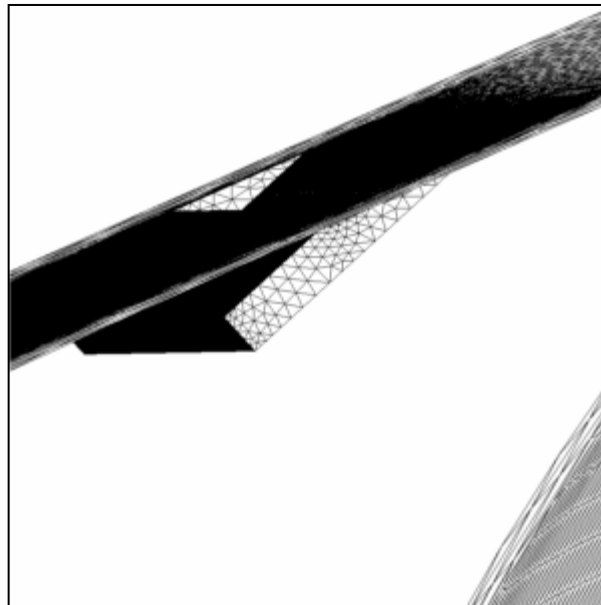
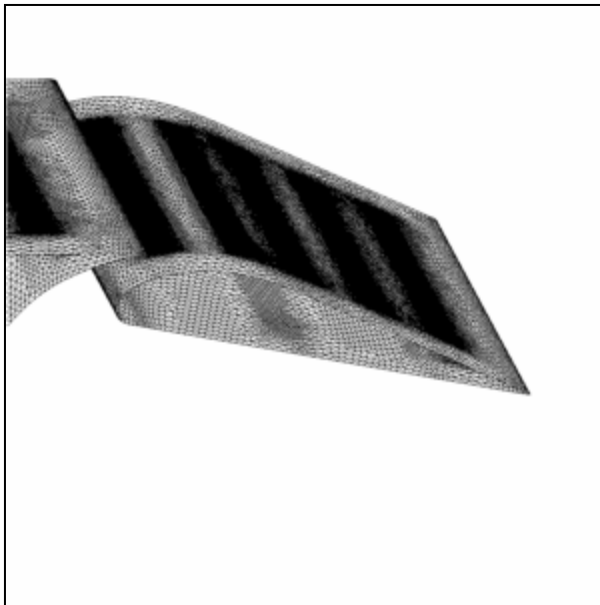


**Shape
Deformation**



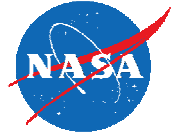
Jet Sliding

**Relative
Translation
And
Rotation**



Jet Incidence

Linearizations of Surface Mesh

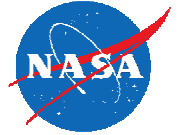


- Rather than develop an analytic form of $\partial \mathbf{X} / \partial \mathbf{Dl}_{surf}$ for the adjoint formulation, these terms are evaluated numerically using complex variables:

$$\frac{\partial f}{\partial x} = \frac{\text{Im}[f(x + i\varepsilon)]}{\varepsilon} + O(\varepsilon^2)$$

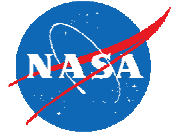
- The procedure used to evaluate the surface mesh is simply wrapped with a loop over the number of design variables:
 - The imaginary part of the current design variable is perturbed by 10^{-50}
 - Complex-valued surface mesh is evaluated as usual
 - Elements of $\partial \mathbf{X} / \partial \mathbf{Dl}_{surf}$ are extracted as the imaginary parts of the result
- The cost of this procedure scales linearly with the number of design variables, but the cost of evaluating the surface mesh is trivial so the overall cost is still negligible

Parallel Scaling and I/O Performance



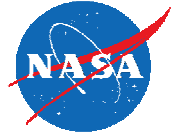
- For optimization of unsteady problems to become routine, massively parallel computing environments must be used as efficiently as possible
- Ideally, execution time will scale inversely with the number of processors used
 - When execution time is dominated by FLOPs, this is generally straightforward with sufficiently large grids on dozens, hundreds, or even a few thousand processors
- For unsteady flows requiring many physical time steps, the use of thousands of processors becomes attractive in addressing extremely long runtimes
- However, partitioning the spatial mesh for these environments can result in exceedingly small amounts of computational work per processor and communication overhead will start to dominate
 - A solver may show linear (or even superlinear) speedup over an initial range of processors, but will generally reach a point of diminishing returns, or even exhibit an inverse scaling behavior

Parallel Scaling and I/O Performance



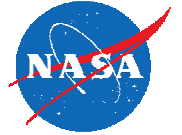
- In the case of time-dependent adjoint solutions, parallel efficiency is further complicated by the need for frequent disk I/O involving large amounts of data
 - Flow solver must write $\mathbf{Q}^n$ to disk at each time step
 - Adjoint solver must read $\mathbf{Q}^n$ from disk at each time step
- This disk I/O can be prohibitively expensive if implemented in a naïve fashion

Parallel Scaling and I/O Performance



- When a single plane of a steady solution or several back planes of an unsteady solution must be written to disk, the traditional strategy in FUN3D is to write local contributions to a single global file from every processor in one of two ways:
 - MPI-IO layer available in MPI-2 standard
 - Native implementation based on stream I/O introduced in Fortran 2003 standard
- Both approaches perform well and a restart file for 20-30 million grid points can be written from $O(500-1,000)$ CPU's in 2-3 minutes using commodity hardware
- This is acceptable when I/O is performed infrequently; cost is amortized over the entire execution
- However, this is completely unacceptable when such data must be stored/loaded at **every** time step

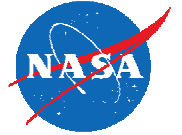
Parallel Scaling and I/O Performance



Original Implementation (2009)

- Each processor writes to (reads from) its own individual direct-access file
 - Direct-access allows file pointer to be placed precisely at the record of interest and I/O may proceed immediately
 - Direct-access method is ~2 orders of magnitude faster than traditional sequential-access approach
- This worked well for cases using 1,024 processors and a Lustre-based parallel file system
- Recent experiments suggest this strategy does not scale well as more processors are used

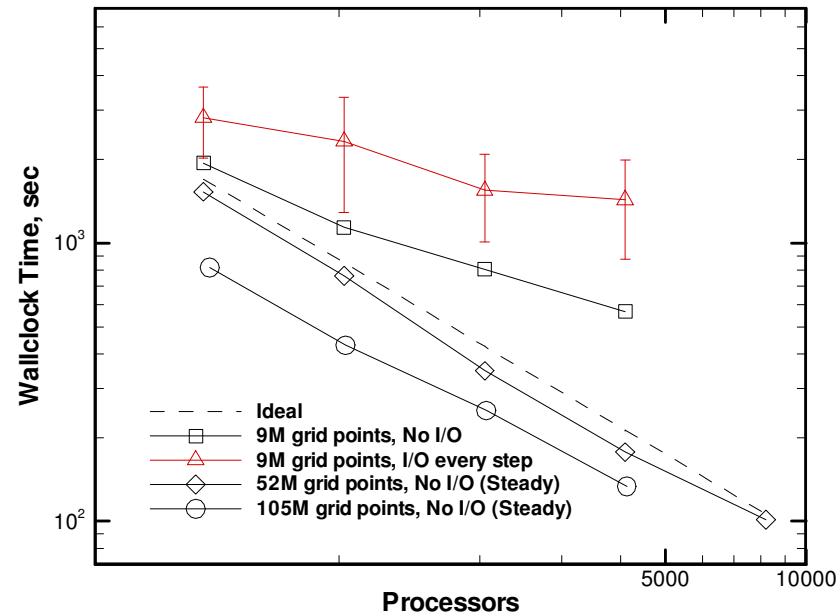
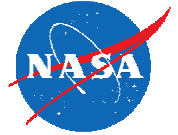
Parallel Scaling and I/O Performance



Test Case

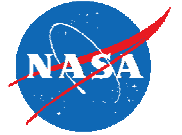
- Two hundred physical time steps of a solution are performed for the current high-lift test case using a range of processor counts
- Each computation is performed fully-dense on dual-socket, quad-core Intel Harpertown processors
- Each case is executed five times to establish repeatability and the effects of varying file system load

Parallel Scaling and I/O Performance Flow Solver



- Computation with no I/O scales poorly at higher processor counts
 - Not surprising, with only 4,000 and 2,000 grid points per processor at the 2,048 and 4,096 processor levels, respectively
- If enough work is present, the implementation scales well in the absence of I/O
 - Steady results for 52M, 105M grid points shown
 - Absolute levels cannot be compared, but slopes confirm desired scaling behavior
- Note: execution times for runs without I/O are highly repeatable
- Writing the solution to disk at each time step using the original strategy costs an overhead and scales poorly
 - Repeatability is erratic; inverse scaling behavior possible

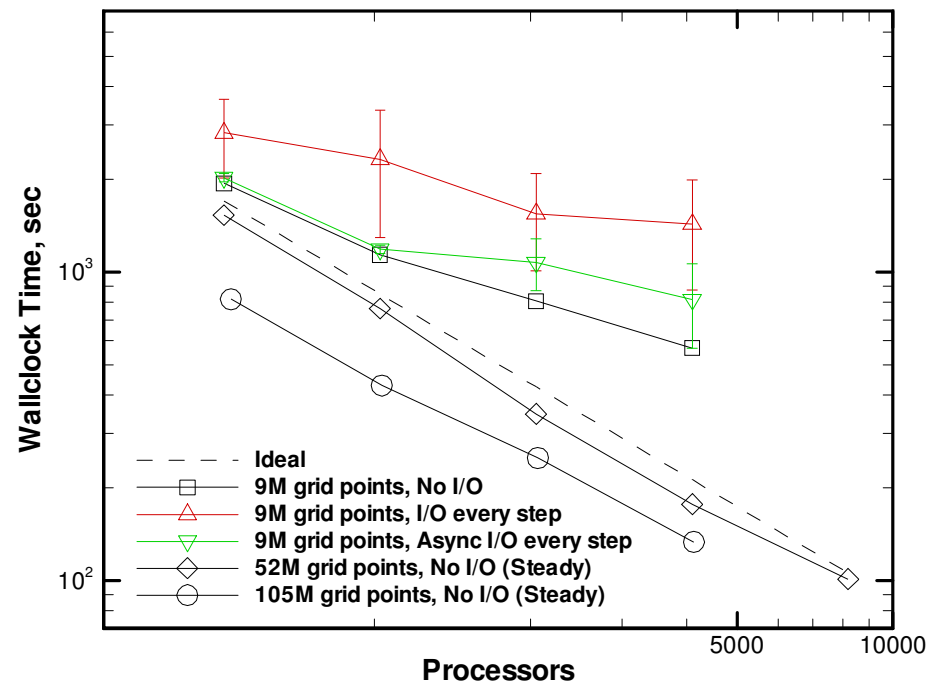
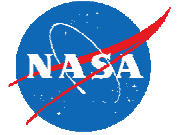
Parallel Scaling and I/O Performance



New Implementation (2010)

- Very similar to original 2009 implementation; however, asynchronous I/O introduced in Fortran 2003 standard is now used
- Data associated with a read or write statement is now buffered, while execution continues for subsequent lines of code
- Strives to “hide” the cost of doing I/O behind the FLOPs
- Care must be taken not to alter the relevant data before the actual I/O has occurred

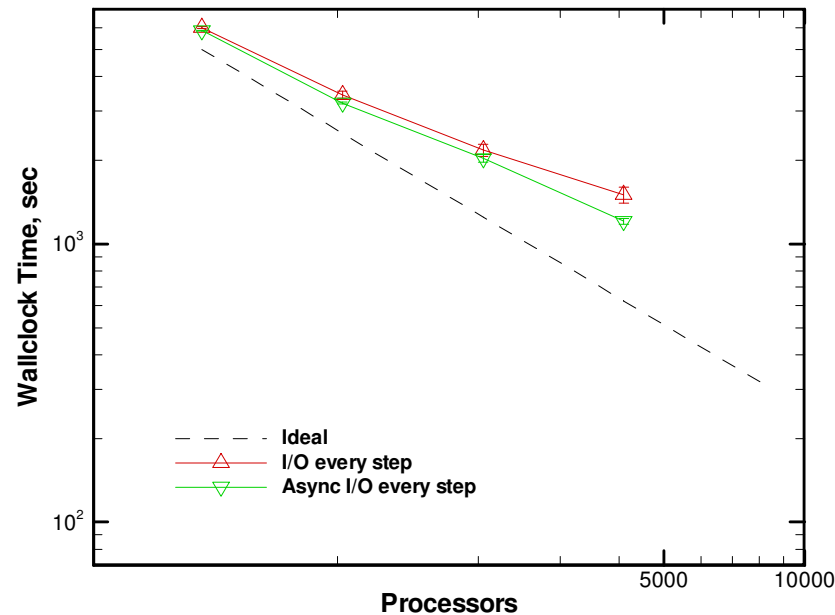
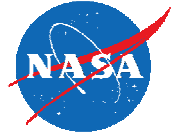
Parallel Scaling and I/O Performance Flow Solver



- At the lower processor counts:
 - I/O appears to have been masked
 - Execution times much more repeatable now
- At the higher processor counts:
 - The asynchronous I/O helps, but still exhibits some overhead
 - Scaling behavior is marginally improved

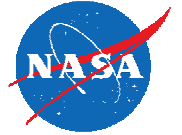
Parallel Scaling and I/O Performance

Adjoint Solver



- Adjoint solution procedure requires Q^n , so a result based solely on computational work is not practical in this case
- Baseline I/O paradigm scales moderately well and is very repeatable
- New asynchronous approach shows improved scaling and slightly better repeatability
- In general, the unsteady adjoint solver shows better performance than the unsteady flow solver when intense I/O is necessary

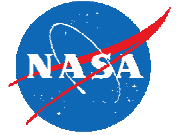
Parallel Scaling and I/O Performance



Possible Reasons Adjoint Solver Outperforms Flow Solver

- Adjoint solver requires considerably more computational work at each time step
 - Spatial residual must be linearized exactly
 - Linear residual during each relaxation sweep requires matrix-vector products over full stencil – typically 4-5 times that of the nearest-neighbor stencil
 - The exact linearizations with respect to the grid must be evaluated
- Flow solver has been extensively optimized for computational performance; adjoint implementation not nearly to the same extent
- File-locking necessary during write operations may also be to blame – no such locking required during read operations

Design Results

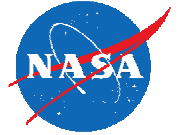


- All cases performed using 256 dual-socket, quad-core Harpertown nodes in fully-dense fashion – total of 2,048 processors
- Nominal wall-clock time to perform and store a flow solution is 7.5 hours
- Approximately 2.1 terabytes of disk space required to store a flowfield solution for all time levels
- Nominal wall-clock time to perform an adjoint solution and sensitivity analysis is 14 hours
- Objective function for all test cases is to maximize the lift coefficient over the final 1000 time steps of the simulation:

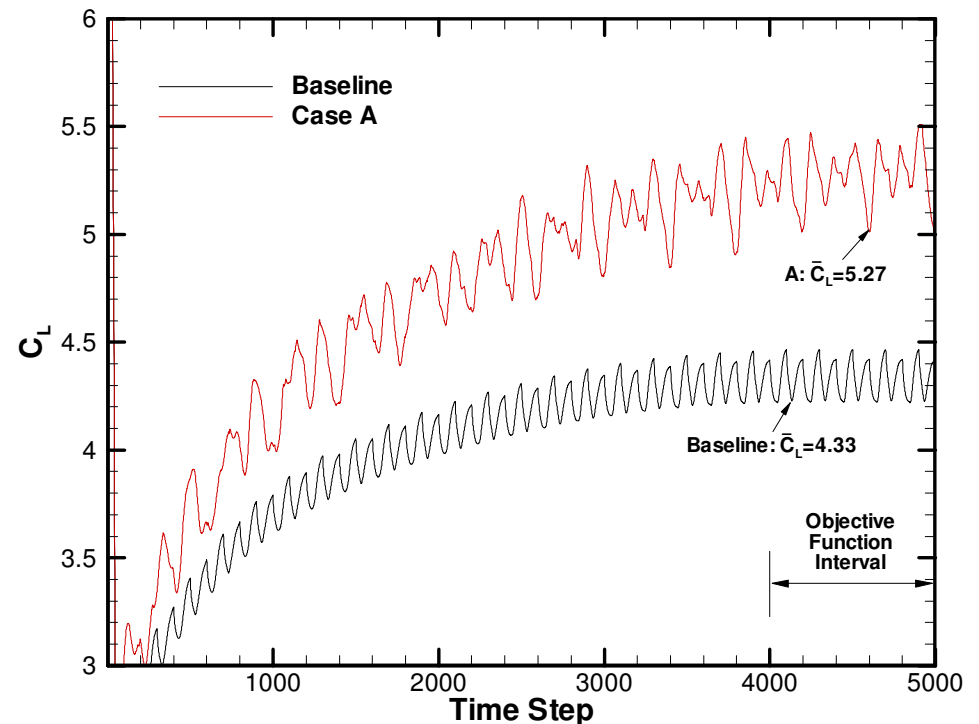
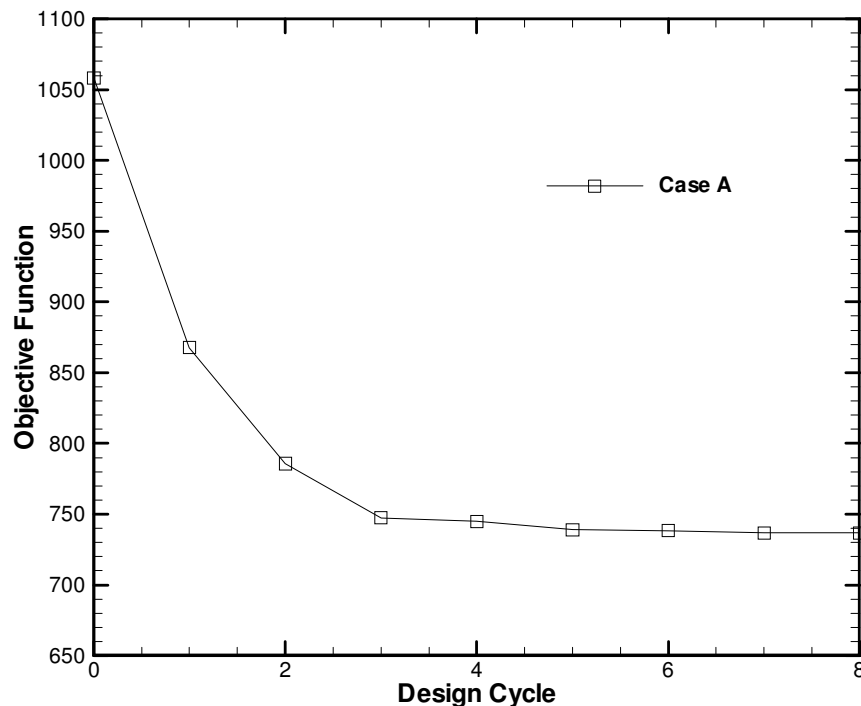
$$f = \sum_{n=4001}^{5000} (C_L - 10.0)^2 \Delta t$$

- All optimizations performed using PORT (Bell Labs)
 - Occasionally requires restarting due to queuing limitations or sporadic system failures
- Three test cases presented
 - Case A: Jet locations, incidence angles, and blowing parameters are active variables
 - Case B: External wing shape also allowed to change
 - Case C: Examines effects of design variable scaling

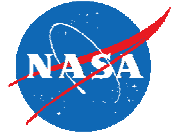
Design Results: Case A



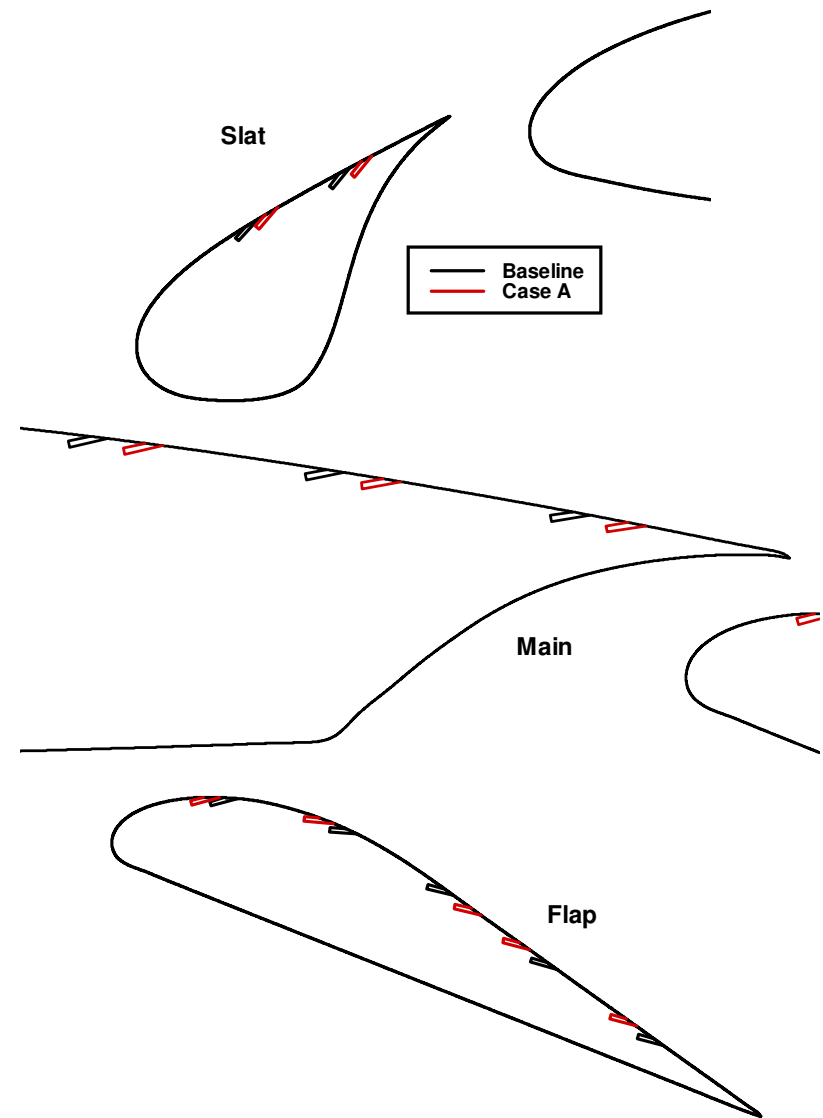
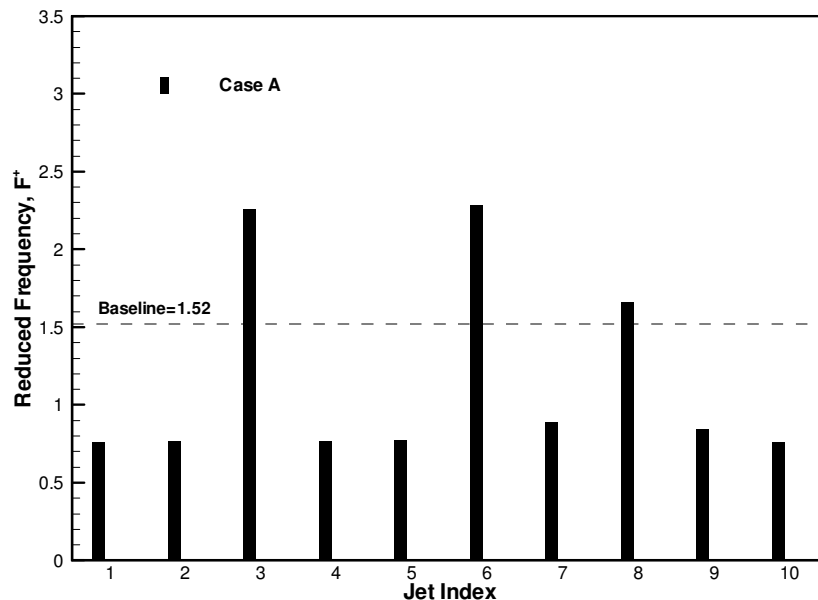
- Five design variables for each jet (total of 50 variables): parametric location and incidence angle; blowing frequency, velocity, and phase shift
- External wing shape held fixed
- Mean value of lift raised 22% from 4.33 to 5.27
- Optimization required 14 flow solutions, 8 adjoint solutions
- Total of 444,000 CPU hours, or 9 days of wall-clock time



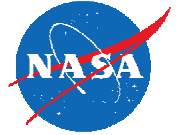
Design Results: Case A



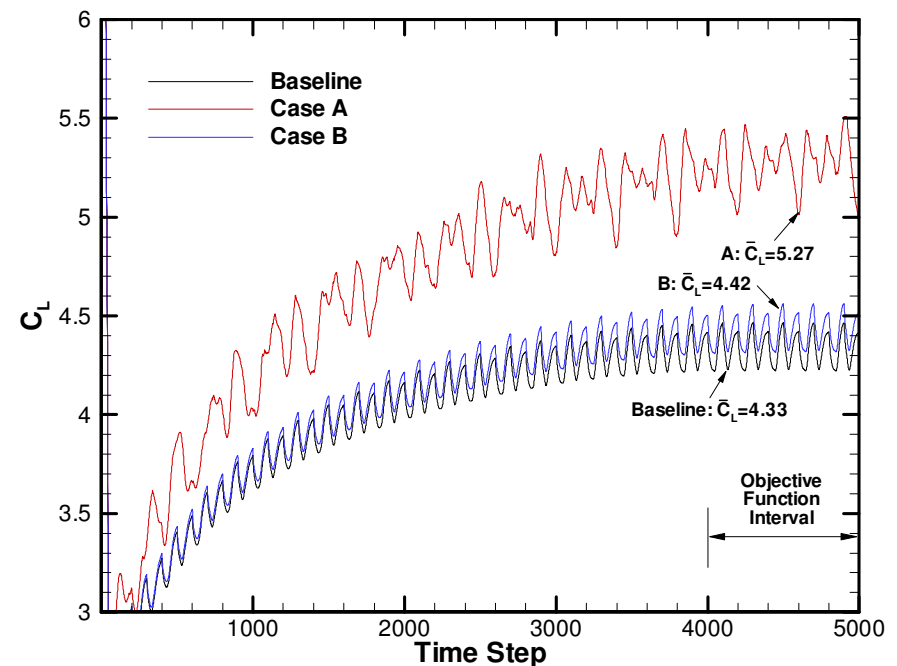
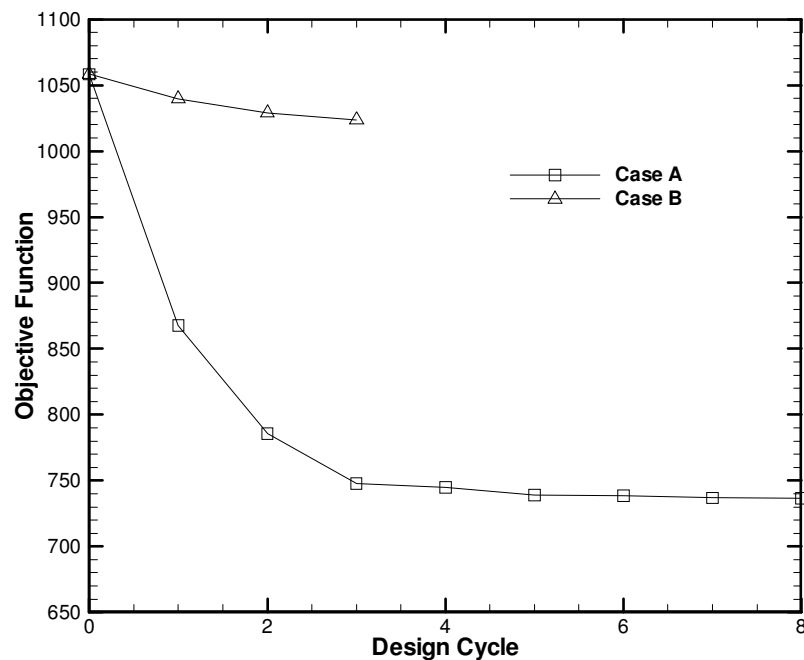
- Jet velocity amplitude increased to its maximum value of 0.48 for all jets
- Most jet frequencies at or near lower bound of 0.76; three raised slightly
- Final phase values all within 0.1° of baseline
- All jets on slat and main elements moved downstream; flap jets have all been moved upstream with exception of jet 8
- All jet incidence angles rotated slightly closer to the surface tangent direction



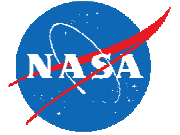
Design Results: Case B



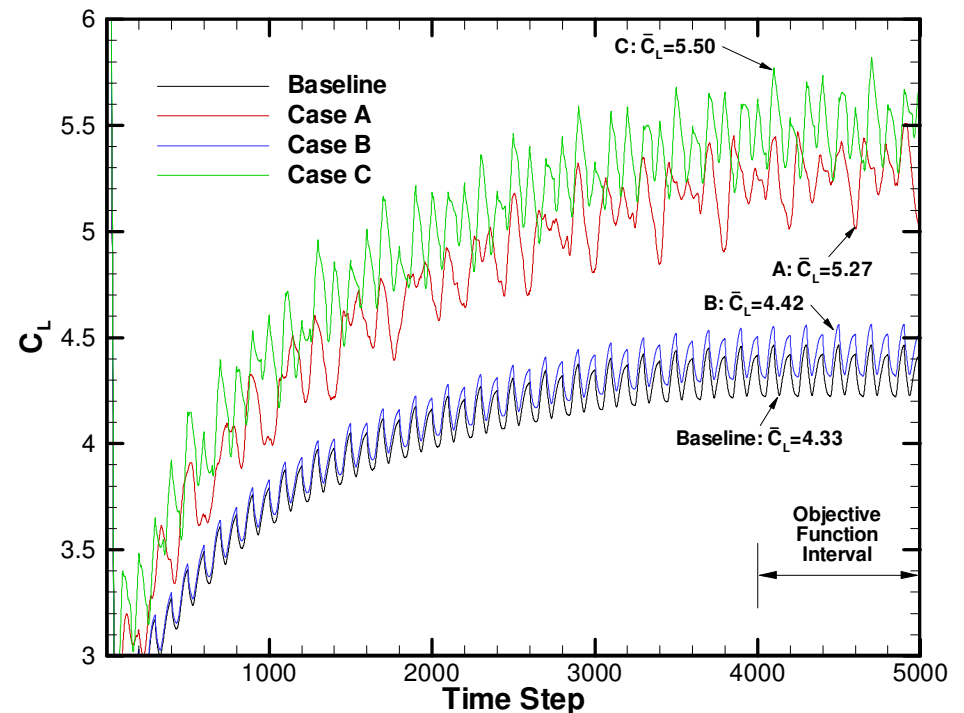
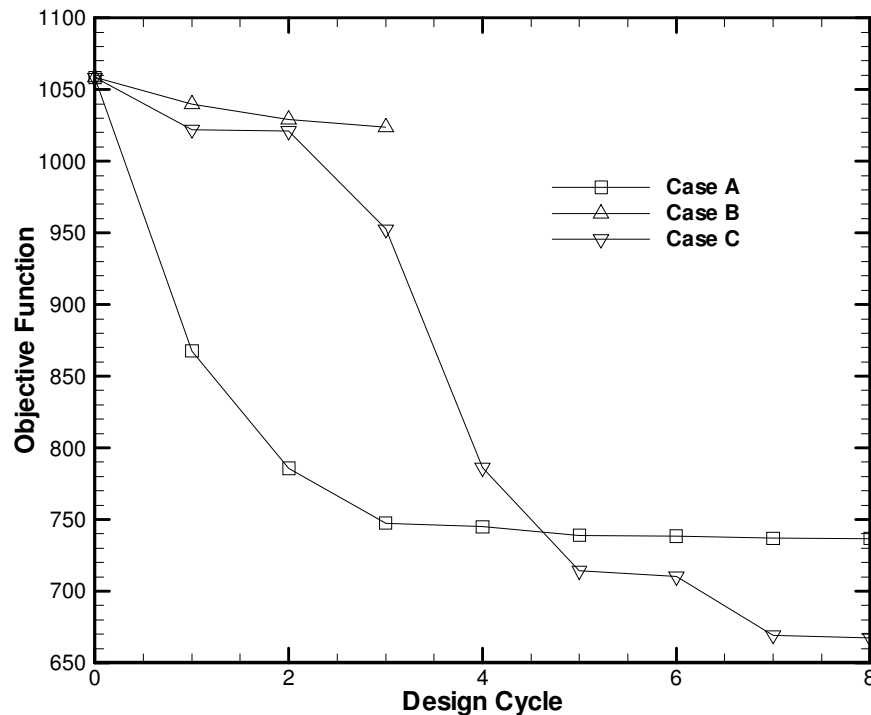
- Same design variables as Case A, **plus**: B-spline control points on all three elements, slat/flap x/z-translations and rotation (total of 161 variables)
- Optimization progresses quite slowly and has difficulty reducing objective function a substantial amount – execution terminated
- Mean value of lift raised 2% from 4.33 to 4.42; changes minor and not shown
- Closer inspection showed optimizer was trying some variables but ignoring others – sensitivity derivatives varied by several orders of magnitude
- Optimization required 8 flow solutions, 3 adjoint solutions
- Total of 209,000 CPU hours, or 4 days of wall-clock time



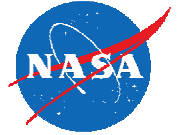
Design Results: Case C



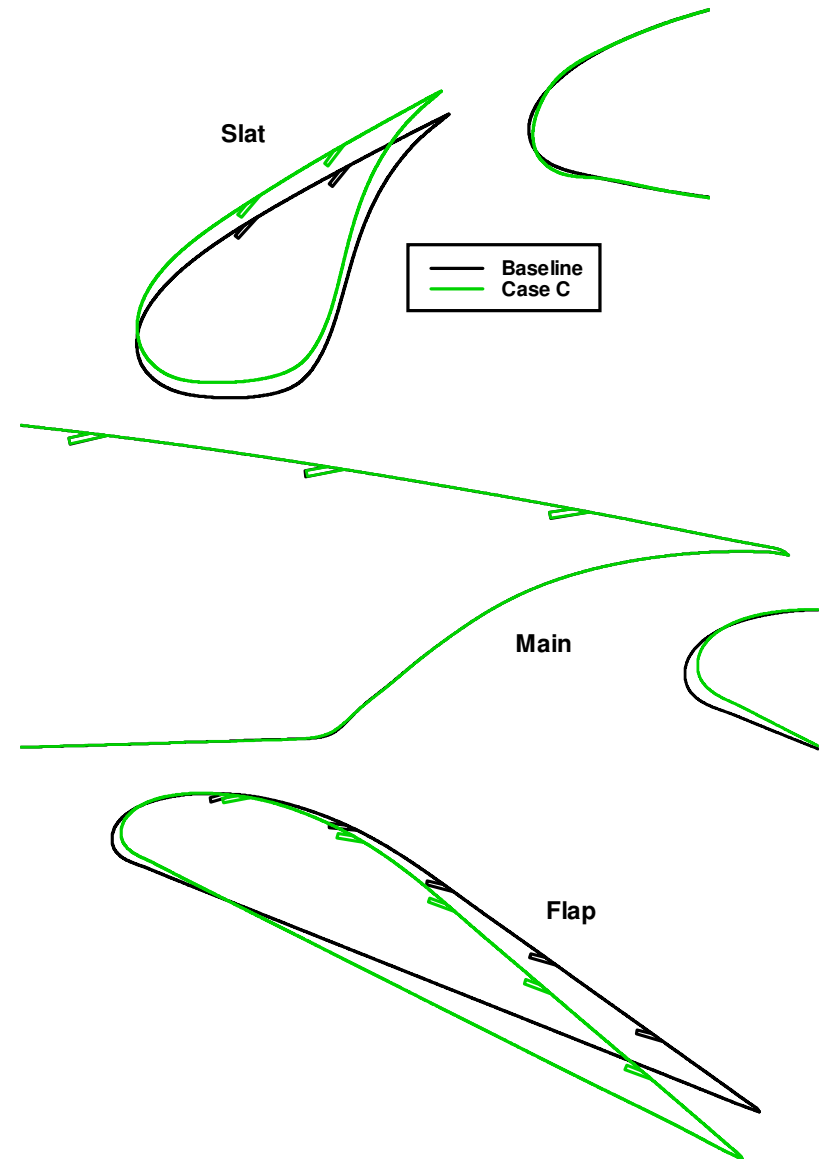
- Same design variables as Case B
- Diagonal scaling now applied to all variables
- Mean value of lift raised 27% from 4.33 to 5.50
- Optimization required 10 flow solutions, 8 adjoint solutions
- Total of 383,000 CPU hours, or 8 days of wall-clock time



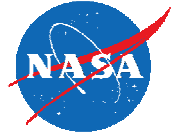
Design Results: Case C



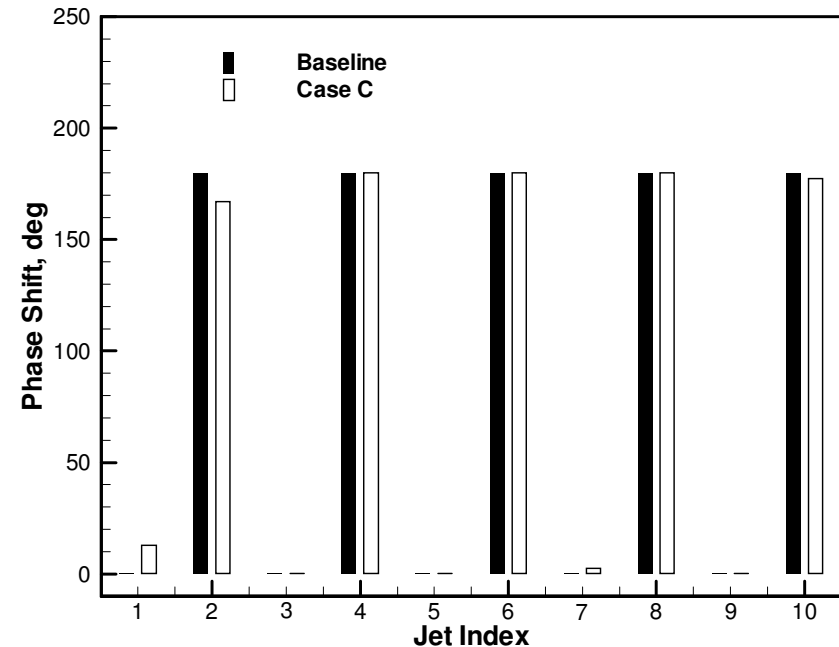
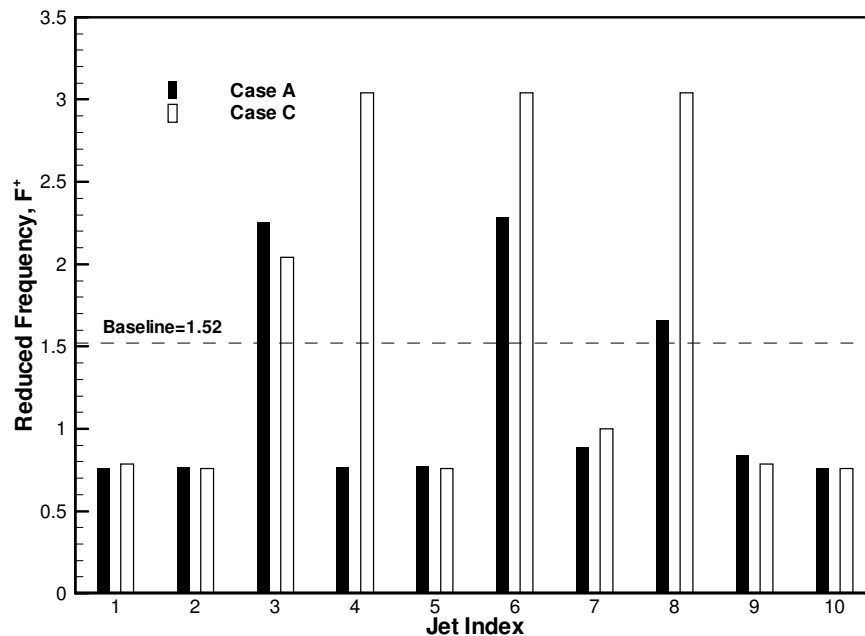
- Jet velocity amplitude increased to its maximum value of 0.48 for all jets, except jet 1, which has been reduced to 0.0
- All jet incidence angles rotated slightly closer to the surface tangent direction
- All jets on slat and main elements moved slightly downstream; flap jets have all been moved upstream with exception of jet 6
- Slat translated upwards and rotated 1.8° nose-down, with slight shape modifications
- Leading edge of main element blunted and slightly thickened
- Flap has been deflected the maximum value of 5° nose-up, translated slightly upwards



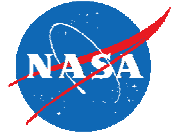
Design Results: Case C



- Final jet frequencies similar to Case A; several much higher
- Phases for jets 1 and 2 show slight changes over baseline values



Summary and Concluding Remarks



- Use of a time-dependent discrete adjoint technique evaluated for optimization of a high-lift wing section with active flow control
 - Design variables included jet configuration, blowing parameters, external wing shape
 - Use of an asynchronous I/O paradigm helpful in alleviating bottlenecks associated with massively parallel file operations
 - Three optimization examples performed – mean value of lift increased as much as 27%; scaling shown to improve numerical conditioning of the optimization procedure
-
- Local-in-time techniques should be pursued to avoid storing the entire solution
 - Higher-order temporal schemes may drastically reduce the number of physical time steps required
 - Adjoint implementation has room for optimization
 - Judicious use of GPU's and other hybrid platforms for massive thread parallelism
 - Continuing to look at massively parallel architectures with 10^4 - 10^5 cores
 - Fault-tolerant algorithms – at application or system level – will become critical with increased levels of parallelism and longer execution times

Acknowledgments

Arvin Shmilovich (Boeing), Chris Rumsey and
Natalia Alexandrov (NASA LaRC), and Boris Diskin (NIA)