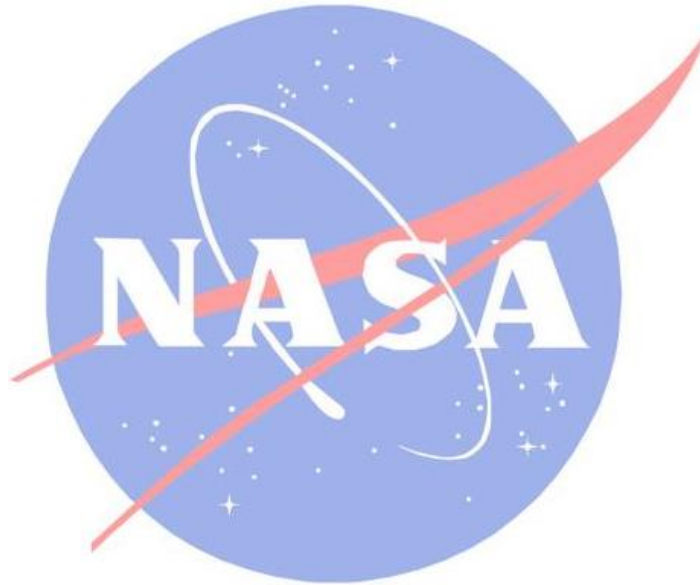




Transonic Aeroelastic Flutter Computations with Mesh Adaptation



Bret Stanford, Steve Massey, Kevin Jacobson, Pawel Chwalowski
NASA LaRC, Aeroelasticity Branch



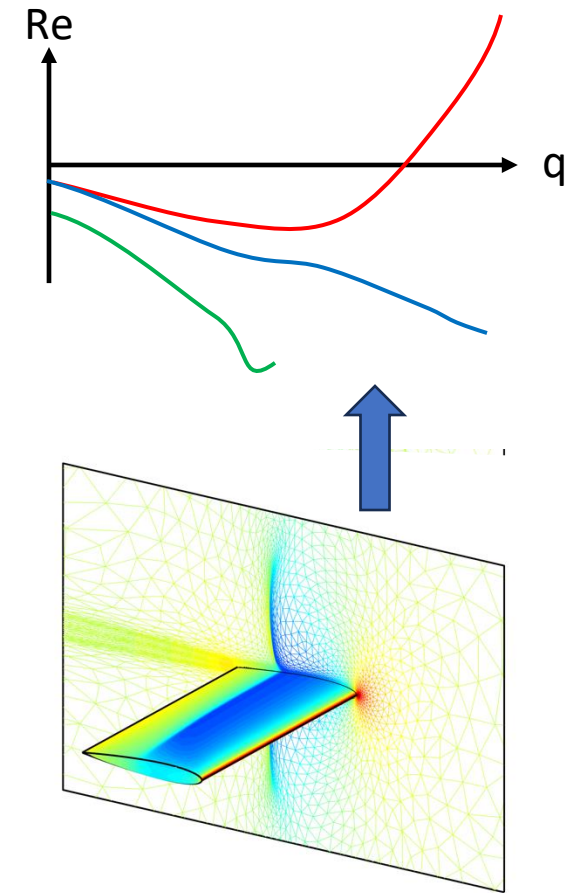
- Two tools used for computational aeroelasticity at NASA LaRC

1. Linearized Frequency Domain (LFD)

- Written into the SFE solver in FUN3D, for either Euler or RANS
- Oscillatory linearizations (via structural mode shapes) about a steady background flow
- Produces the Generalized Aerodynamic Forces (GAFs), which are then used to compute flutter-q

2. Mesh refinement

- *refine* (<https://github.com/nasa/refine>): multiscale mesh refinement
- *pyrefine* (<https://github.com/nasa/pyrefine>): python wrapper of *refine*, that allows us to integrate LFD
 - Can track flutter-q vs. the size of the CFD mesh





Computational Tools (2)



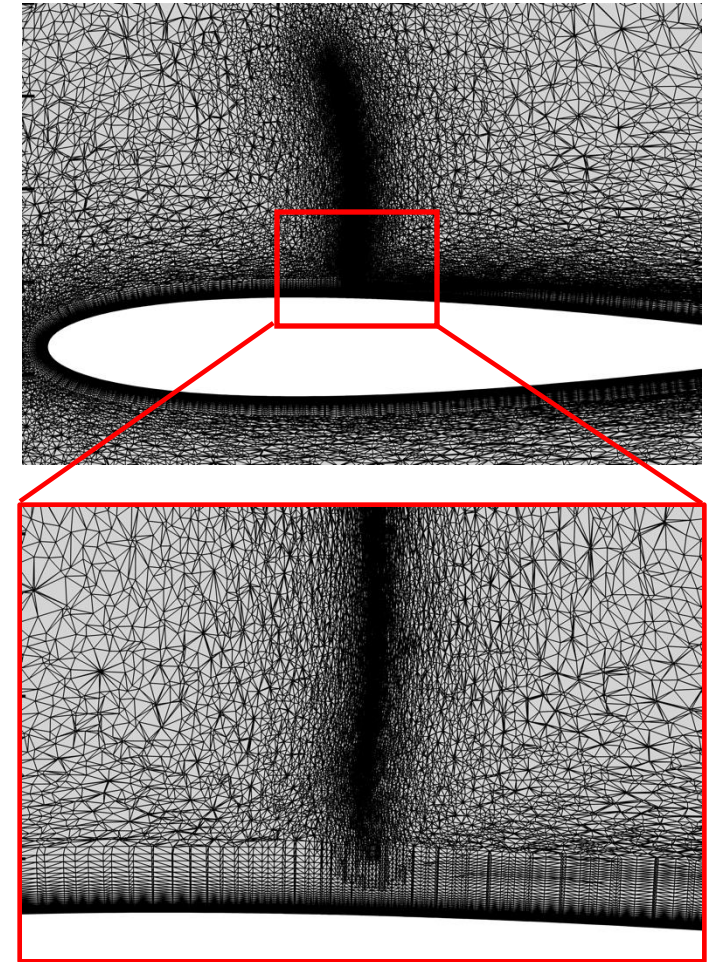
- The LFD + *refine* workflow cannot handle cases with static aeroelastic deformations
 - *refine* creates meshes that are very non-prismatic near the wing surface
 - These meshes cannot be deformed w/o negative volumes
- So our previous work has focused on:
 - Cases without static aeroelastic deformation (AGARD 445.6)
 - Pitch/plunge cases, where static pitch motion can be handled without mesh deformation
- But for any realistic wing configuration, there will be static aeroelastic deformations
 - And for transonic flows, the relationship between dynamic flutter and static aeroelasticity is very important



Mesh Refinement



1. Start with a steady FUN3D-based flow solution
 2. Using the scalar Mach field, *refine* creates a multiscale metric field
 3. Sources from this field are fed to *refine*, which creates the next mesh
 3. Sources from this field are fed to HeldenMesh, which creates the next mesh
- HeldenMesh can create meshes with adapted prismatic cells in the boundary layer
 - Sources in the boundary layer are deleted before being fed to HeldenMesh
 - These meshes can be statically deformed, w/o negative volumes
 - It is possible to do this entire process in HeldenMesh, and not use *refine* at all





Mesh Adaptation Workflow



1. Start with an initial coarse mesh (manually-constructed)
2. Compute a steady flow solution with FUN3D (SFE, RANS, SA-neg)
3. Integrate these pressures over the wing surface, using the structural mode shapes
 - This computes the static generalized aerodynamic forces, GAF
4. Compute each of the modal amplitudes
 - $\eta_i = GAF_i / \omega_i^2$
 - ω_i are the natural frequencies of each mode
5. Deform the volume CFD mesh using these structural deformations
6. Return to step 2, repeat until the static aeroelastic deflections are converged
7. Feed the steady flow field to *refine* / HeldenMesh, to compute the next mesh



Staggered Static Aeroelasticity



- This process is very expensive: requires a complete static aeroelastic convergence for each mesh
- A cheaper idea:
 - Use the steady flow of mesh k to compute modal amplitudes
 - But rather than use those modal amplitudes to deform mesh k , and drive the static aeroelastic residual to 0...
 - We instead move on to mesh $k+1$ right away, and use the modal amplitudes of mesh k to deform mesh $k+1$
- This only requires one FUN3D solve per mesh
- The static aeroelastic solution converges alongside the mesh adaptation process
- No guarantee that this will work, but for the cases shown here, it does mostly work
- Aitken relaxation to speed up the convergence of the static aeroelasticity



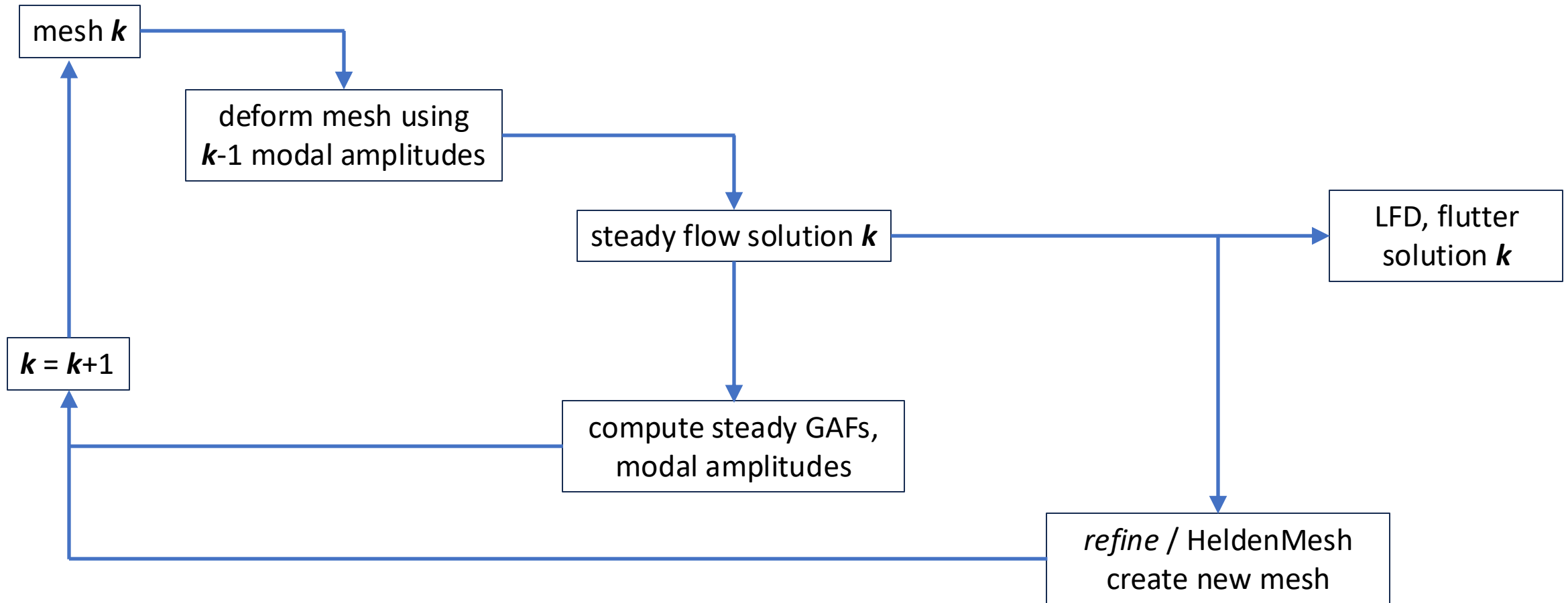
Linearized Frequency Domain



- For a given mesh k , FUN3D computes the steady flow about a statically deflected wing
 - The LFD solver is used to linearize about that flow, and compute the flutter behavior
- Flutter-q can converge alongside both the mesh adaptation process and the static aeroelastic solution
- LFD/flutter computations are only along for the ride, and have no impact on the mesh adaptation process



Final Workflow





Test Case



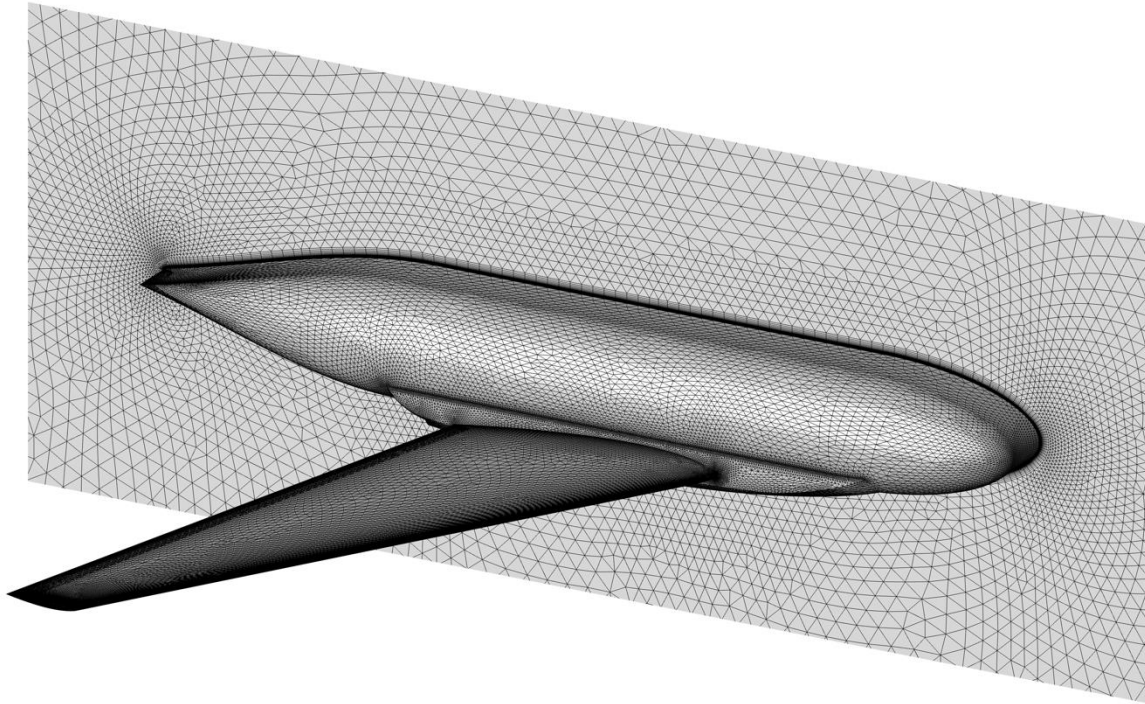
- Semispan flutter model tested numerous times in the LaRC Transonic Dynamics Tunnel (TDT)
 - Semispan of 53", 23 deg. sweep, 0.3 taper ratio
 - Aluminum/foam construction
 - 0012 airfoil
- 6 structural modes included: 4.3 Hz first bending mode, up to 61.4 Hz (second torsion)
- Mesh adaptation:
 - Starts with a baseline 2.5M node mesh
 - 7-8 mesh adaptation steps completed for each case, with finest mesh 11-12M nodes
 - LFD solves become very difficult-to-converge for finer meshes than this, under transonic flows



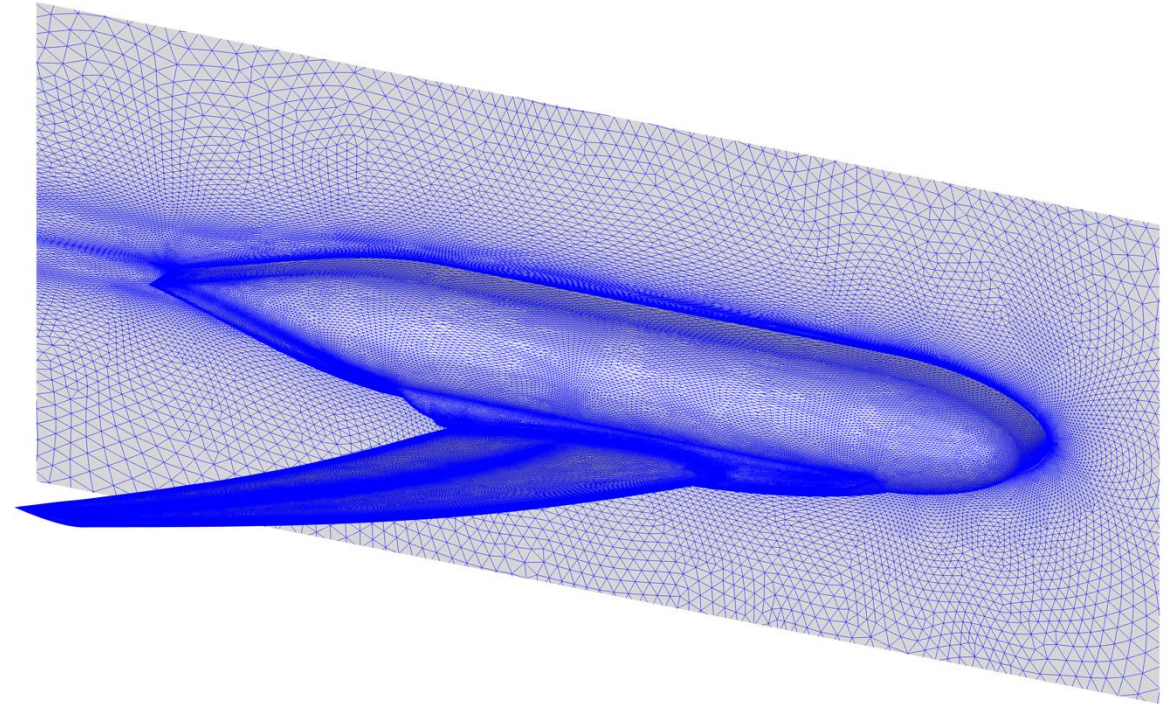
credit: NASA



Mach = 0.8, AoA = 3°, q = 110 psf



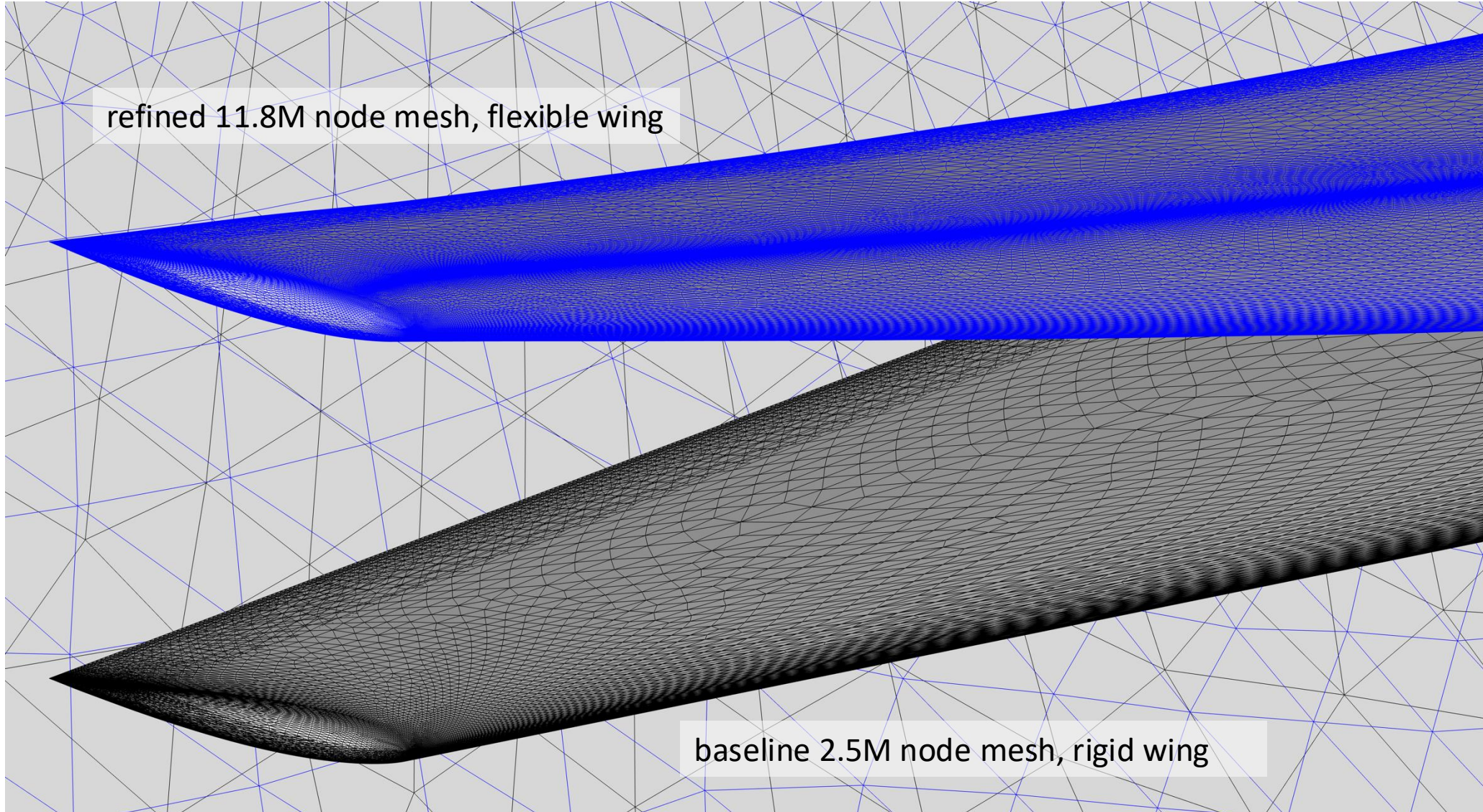
baseline 2.5M node mesh, rigid wing



refined 11.8M node mesh, flexible wing

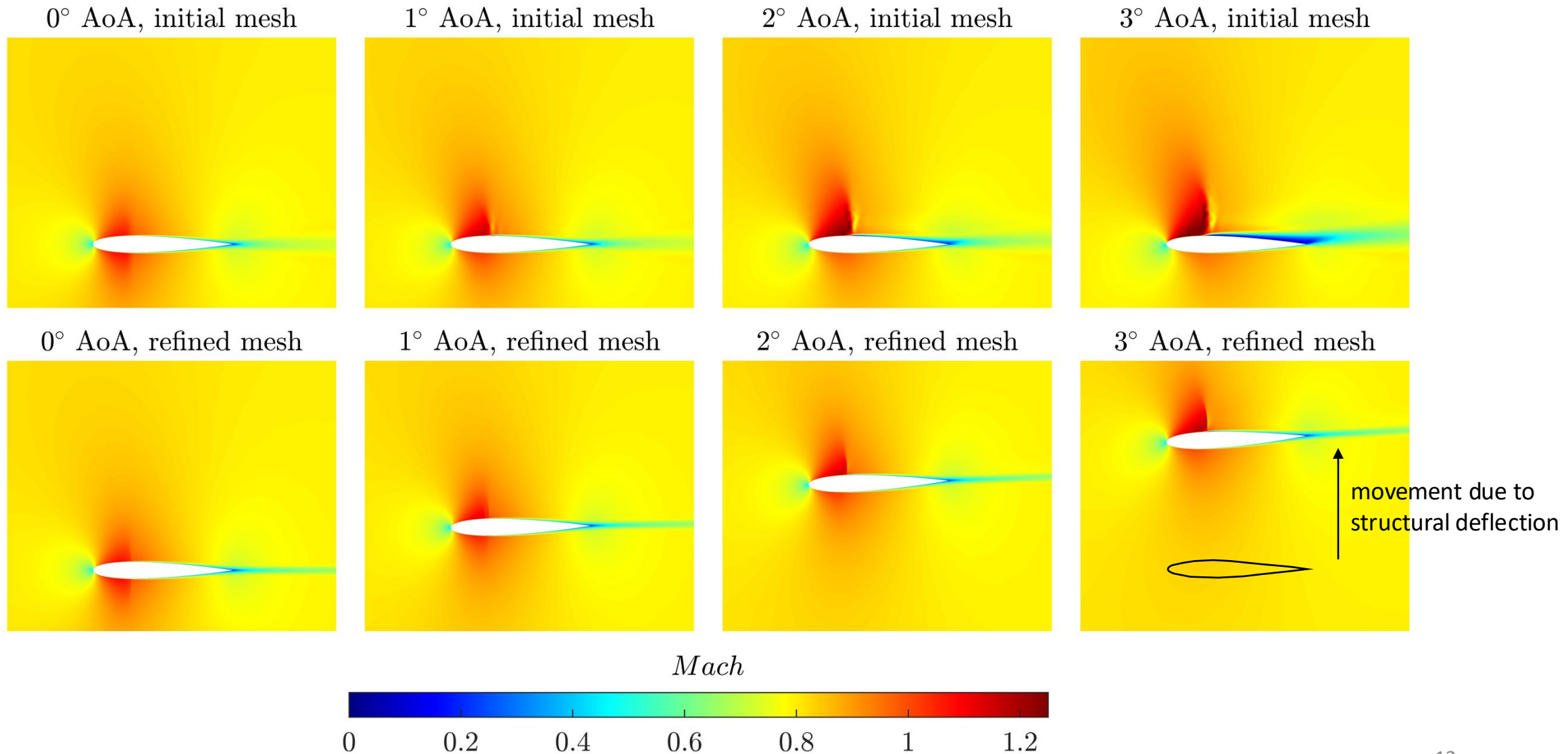


Mach = 0.8, AoA = 3°, q = 110 psf



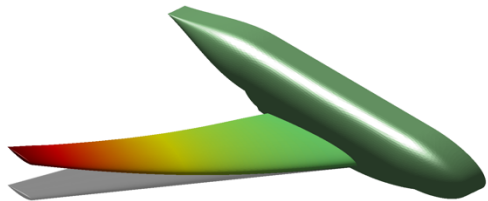


Slices at 95% Semispan, Mach = 0.8

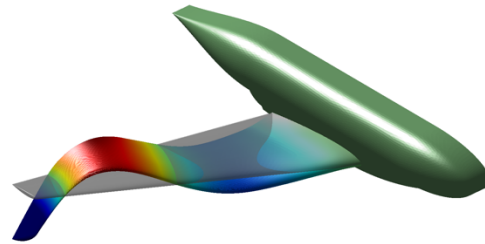
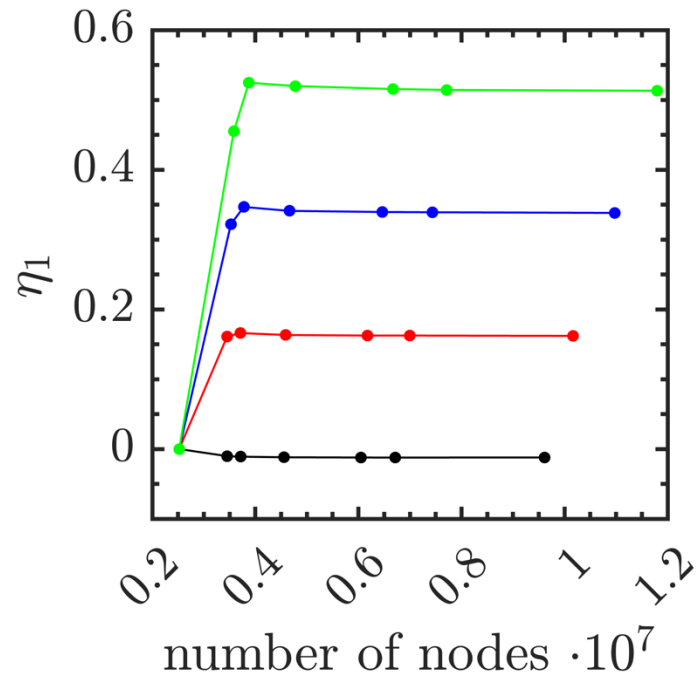




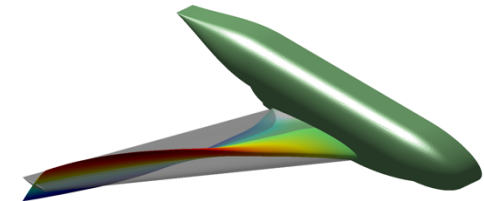
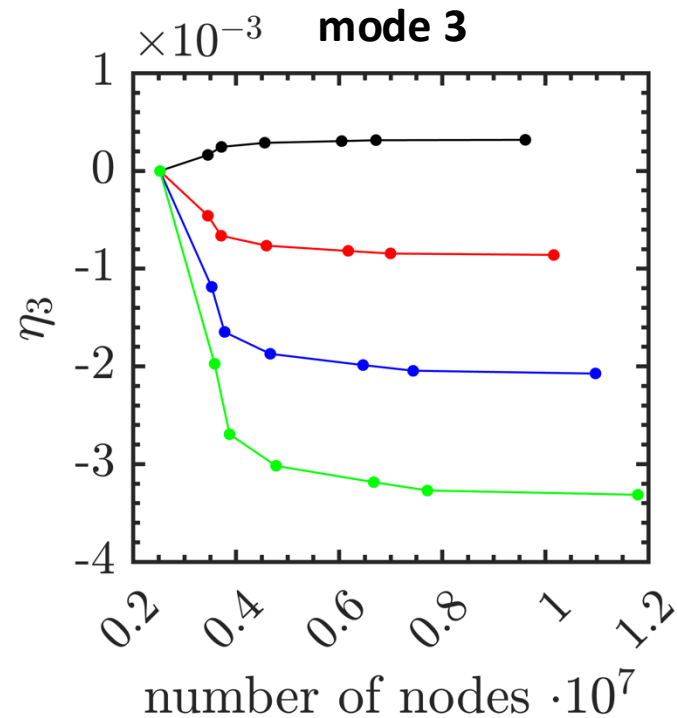
Static Aeroelastic Convergence



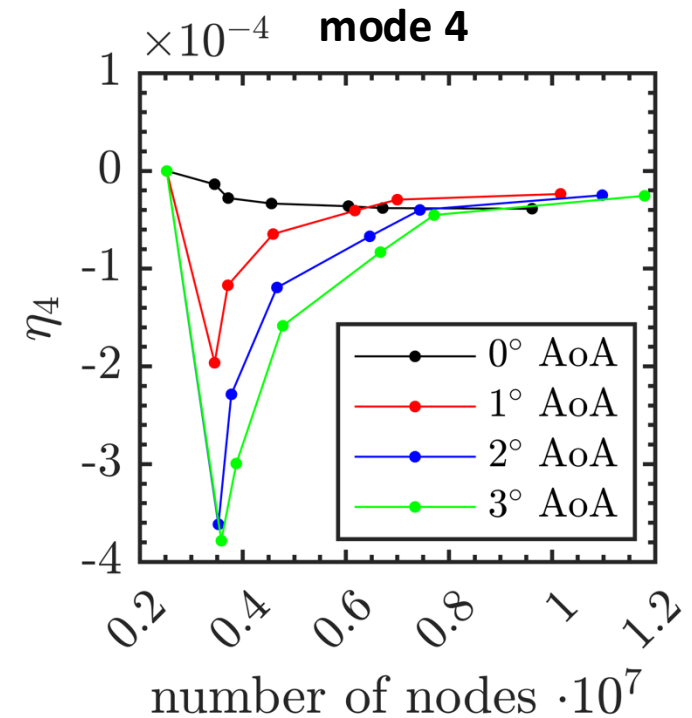
mode 1



mode 3

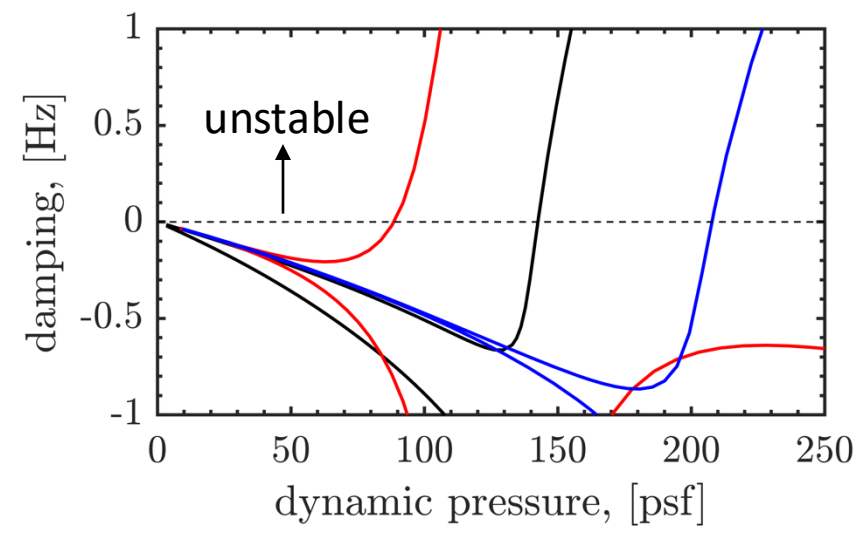
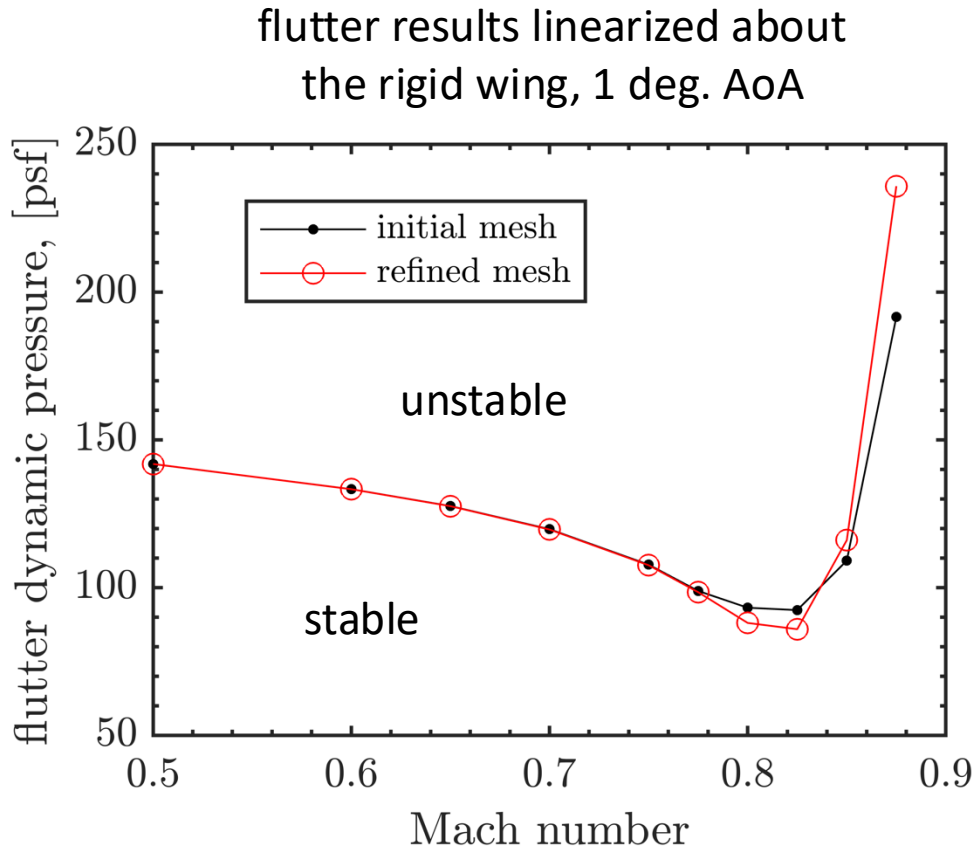


mode 4

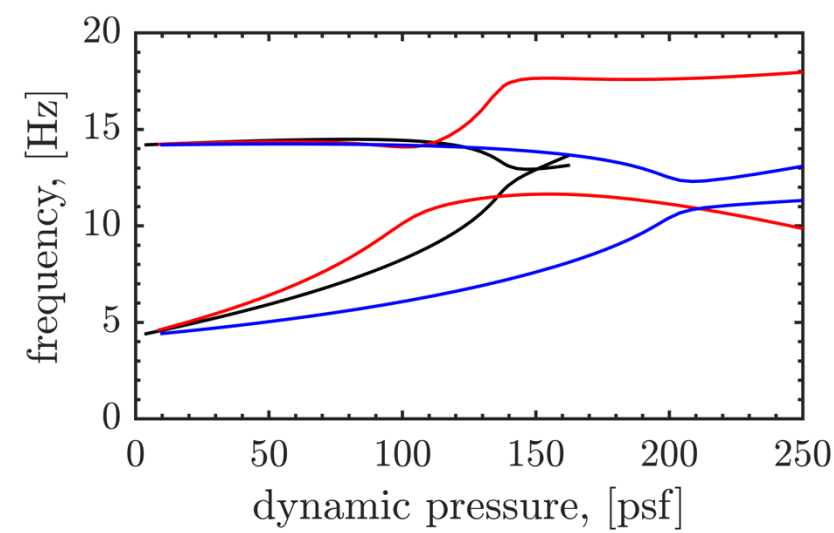




Flutter Results, Mach Sweep



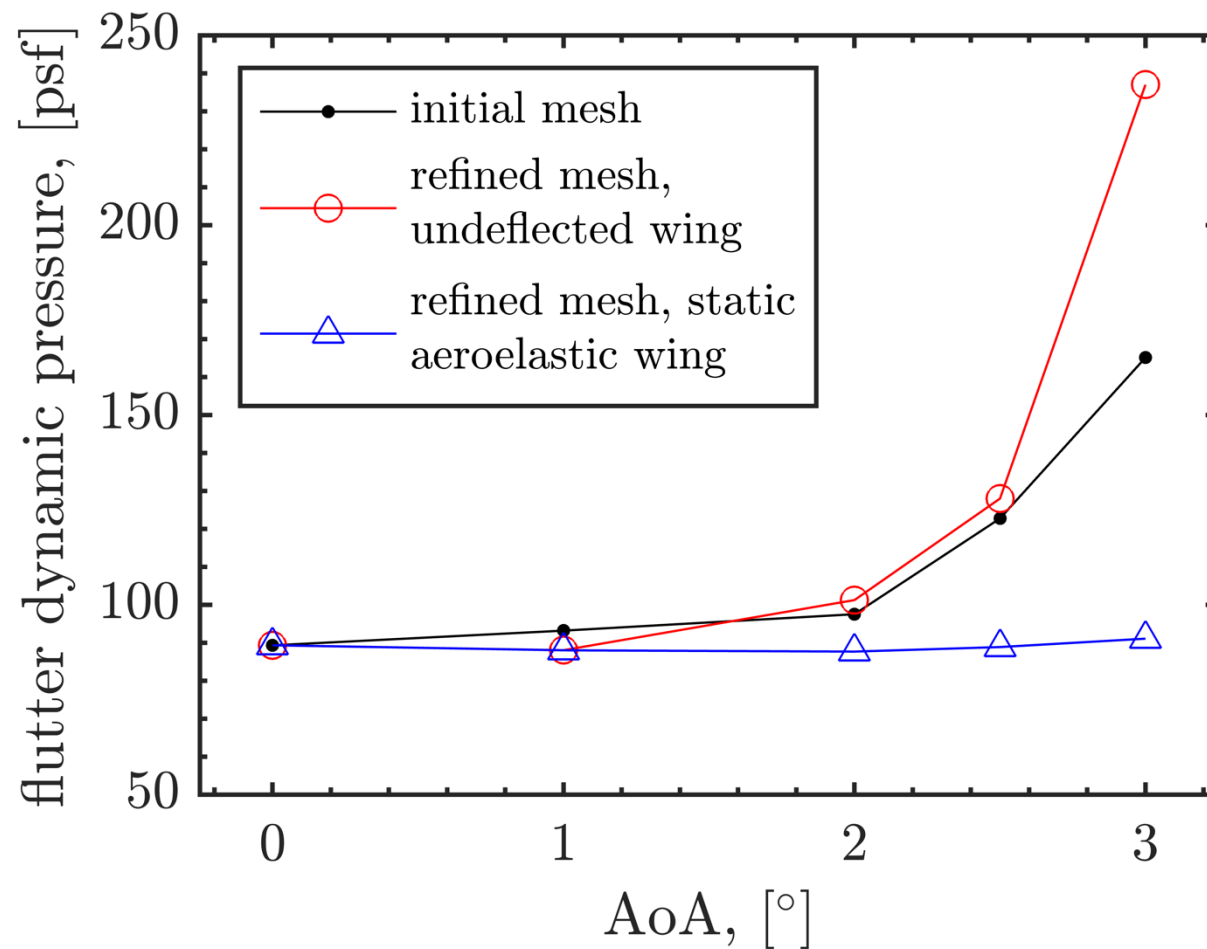
Mach 0.5
Mach 0.825
Mach 0.875



Mach 0.5
Mach 0.825
Mach 0.875



Flutter Results, Mach 0.8



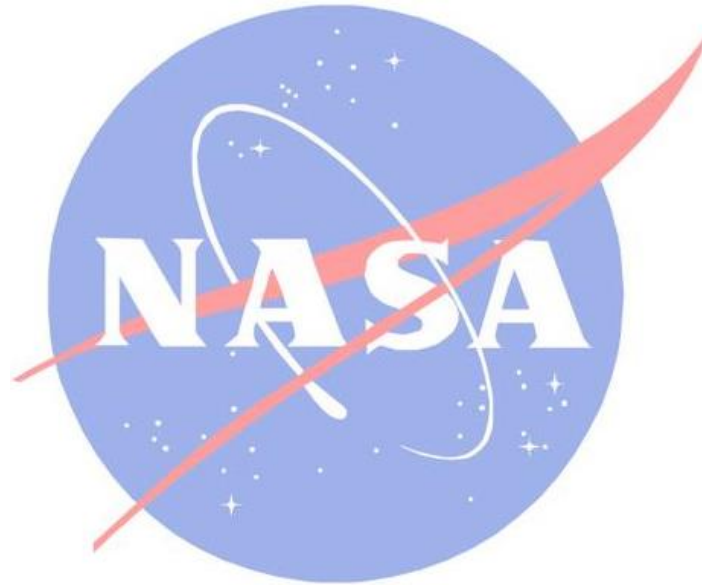
- Large drop in flutter-q due to static aeroelastic deformation
- Adaptive washout of the swept wing reduces the strength of the transonic nonlinearities
- Pushes flutter point from backside to frontside of dip



Conclusions



- Mesh adaptation of a transonic aeroelastic wind tunnel model
- Steady and LFD flow solves with FUN3D/SFE
- Mesh refinement with *refine* and HeldenMesh
 - Workflow generates deformable meshes with a prismatic layer near the surface
- Demonstration cases show mesh adaptation of a static aeroelastic wing shape
 - Static aeroelastic mechanics converge alongside the mesh adaptation process, with a staggered approach
- LFD is also used to compute the flutter mechanism linearized about the static shape
 - For this swept wing, there is a large dependence of transonic flutter upon the static shape
- Paper also shows some cases with matched-point flutter requirements, and some comparison with experimental data



bret.k.stanford@nasa.gov