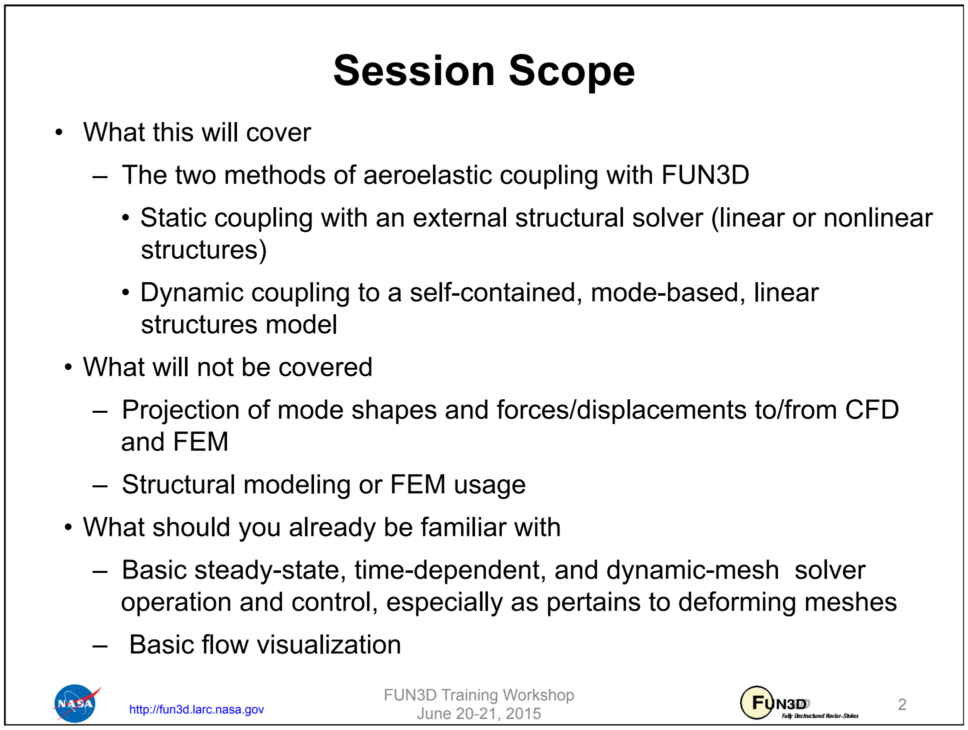
FUN3D v12.7 Training

Session 16: Aeroelastic Simulations

Bob Biedron



Session Scope

- What this will cover
  - The two methods of aeroelastic coupling with FUN3D
    - Static coupling with an external structural solver (linear or nonlinear structures)
    - Dynamic coupling to a self-contained, mode-based, linear structures model
- What will not be covered
  - Projection of mode shapes and forces/displacements to/from CFD and FEM
  - Structural modeling or FEM usage
- What should you already be familiar with
  - Basic steady-state, time-dependent, and dynamic-mesh solver operation and control, especially as pertains to deforming meshes
  - Basic flow visualization

Introduction

- Background
  - Aeroelastic problems of interest that can be tackled with FUN3D fall into 2 general categories
    - Static: structural displacement asymptotes to a fixed level; coupling between CFD and CSD can be done infrequently - typically interested in accounting for the structural displacement on (say) cruise performance
    - Dynamic: the change in aero affects the structural deformation to the extent that there is an unsteady coupling between the two; coupling between CFD and CSD must be done frequently - prediction of flutter onset is the classic example
- Compatibility
  - Compatible with compressible flow; mixed elements; 2D/3D
- Status
  - Modal (flutter) analysis fairly routine; static FEM coupling much less so - still evolving; dynamic FEM coupling needs “framework”

Static Aeroelastics - Overview

- Basic process (*not* a **`moving_grid`** problem - no **`moving_body.input`**)
  1. Solver starts with an initial grid and solution
  2. Solver reads in a new surface shape and deforms the mesh to fit
  3. Solver performs the requested number of iterations, and outputs aerodynamic loads to a file
  4. Middleware maps aerodynamic loads at CFD grid points onto FEM grid
  5. Structural solver computes new displacements from the airloads
  6. Middleware maps structural displacements onto new surface
  7. Back to step 2; repeat until converged - airloads and displacements
- Historically, Jamshid Samareh of NASA Langley provided middleware (“DDFdrive”) for this loads and deflection transfer; release of this software transitioning to FUN3D team – contact FUN3D-Support@lists.nasa.gov
- In principle, the above could be applied every time step of a dynamic aeroelastic case; however, file I/O is very inefficient for this

Static Aeroelastics - New Surface Shape

- Reading of the updated surface(s) is triggered by the CLO **`--read_surface_from_file`**
  - File(s) read once at the start of solver execution (steady-state mode)
  - File root name must be of the form **`[project]_bodyN`** (for body N)
  - File extensions: **`.dat`** or **`.ddfb`**
    - **`[project]_bodyN.dat`** ASCII Tecplot file, “FEPOINT” style
    - **`[project]_bodyN.ddfb`** Binary (“stream”) DDFdrive style
    - DDFdrive middleware supports both - **`.ddfb`** preferred
  - File provides new x,y,z coordinates for each surface point plus an integer that identifies the point in the volume-mesh numbering system
  - Options for this *input* surface file input are specified in the **`&massoud_output`** namelist (details later)

Static Aeroelastics - Aero Loads Output

- Output is triggered by the CLO **`--write_aero_loads_to_file`**
  - File(s) written at a user-controlled frequency
  - File root name of the form **`[project]_ddfdrive_bodyN`** (Nth body)
  - File extensions: **`.dat`** or **`.ddfb`**
    - **`[project]_ddfdrive_bodyN.dat`** ASCII Tecplot file, “FEPOINT” style
    - **`[project]_ddfdrive_bodyN.ddfb`** Binary (“stream”) DDFdrive style
    - DDFdrive middleware supports both
  - File provides current Cp, Cfx, Cfy, Cfz for each surface point plus an identifier that maps the point in the volume mesh
  - Options for this *output* surface file input are specified in the **`&massoud_output`** namelist (next)

Static Aeroelastics - `&massoud_output` (1/2)

- The **`&massoud_output`** namelist serves several closely-related purposes, and the name is not especially well-suited to any of them…
- For static aeroelastics, it is used to
  - Define the aeroelastic body(s) as a collection of boundary surfaces
  - Specify the format of the new surface file and the output aero loads file
  - Specify the frequency of the aero loads output
- Naming convention: **`[project]_ddfdrive_bodyN.dat`**
- Example:

```
&massoud_output
  aero_loads_file_format = 'stream'  (default = 'ascii')
  massoud_file_format    = 'stream'  (default = 'ascii')
  aero_loads_output_freq = -1        (if +n…output every n steps)
  n_bodies               = 1         (default = 0)
  nbndry(1)              = 3         (default = 0)
  boundary_list(1)       = '1,2,3'   (default = '')
/
```

Static Aeroelastics - `&massoud_output` (2/2)

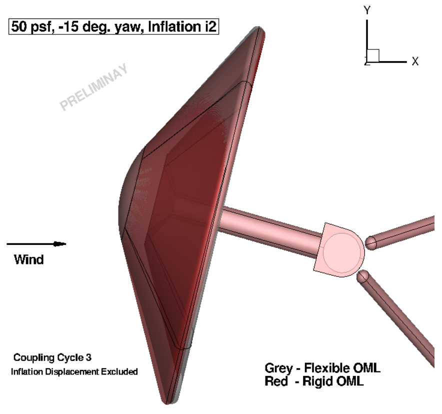
- The **`&massoud_output`** namelist has additional options
  - rotate, translate and scale the geometry written to the aero loads file
  - multiply the aero coefficients by the dynamic pressure to get forces
  - rotate, translate and scale the geometry read from the new surface file
  - output aero loads on either the deflected or undeflected surfaces
  - See Manual for these infrequently-used options

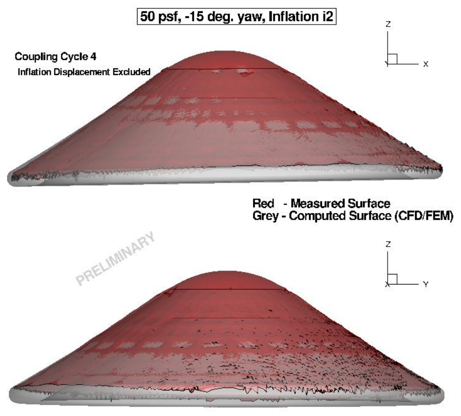
Static Aeroelastics - Towards Automation

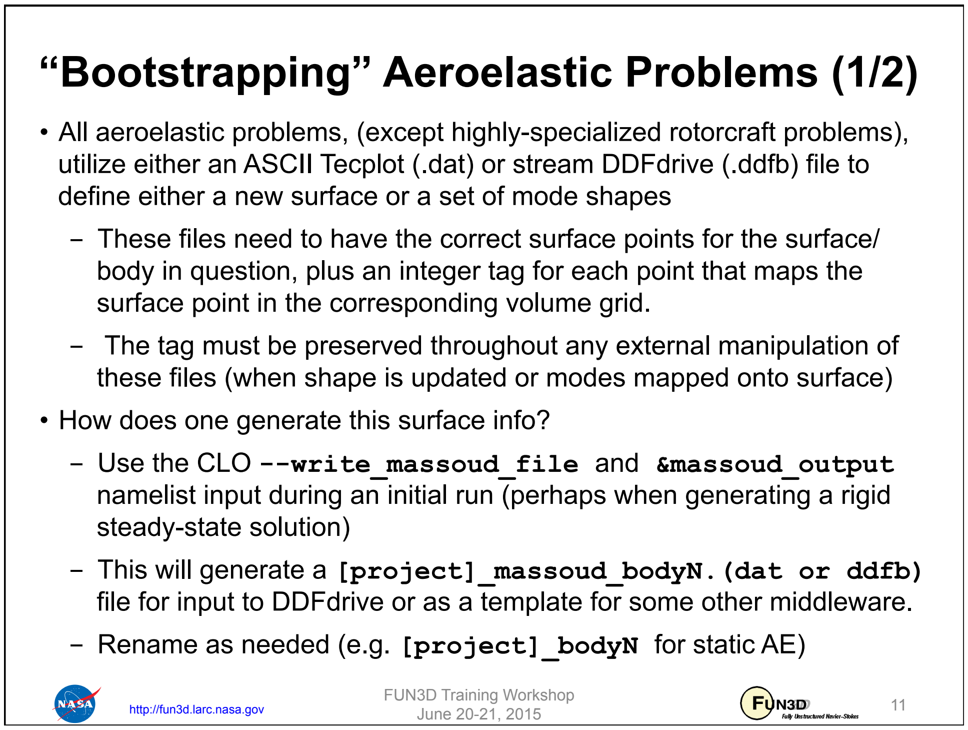
- As outlined, the process is rather cumbersome, with multiple separate steady-state runs of the flow solver, the FEM and middleware
  - For our own in-house work we have written a few scripts to orchestrate these steps using DDFdrive
  - Contact FUN3D-Support@lists.nasa.gov if interested (scripts are not part of standard distribution)
- Longer term, we plan on a "framework" to allow direct access of data between CFD, FEM and middleware to avoid file I/O
  - Helpful for static coupling with an FEM; essential for dynamic coupling
  - Stay tuned for development along these lines

Static Aeroelastic Coupling Example

Recent Application: Inflatable Decelerator - Low Speed Test



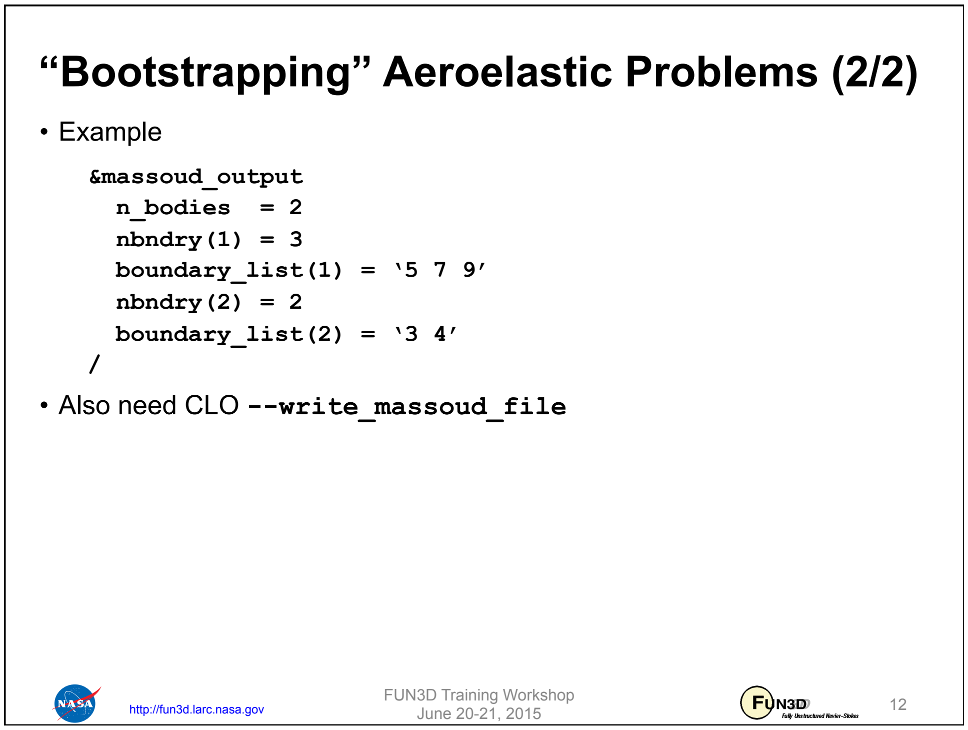

50 psf, -15 deg. yaw, Inflation i2
Coupling Cycle 4
Inflation Displacement Excluded
Z
X
Red - Measured Surface
Grey - Computed Surface (CFD/FEM)
PRELIMINARY
Z
Y



“Bootstrapping” Aeroelastic Problems (1/2)

- All aeroelastic problems, (except highly-specialized rotorcraft problems), utilize either an ASCII Tecplot (.dat) or stream DDFdrive (.ddfb) file to define either a new surface or a set of mode shapes
  - These files need to have the correct surface points for the surface/ body in question, plus an integer tag for each point that maps the surface point in the corresponding volume grid.
  - The tag must be preserved throughout any external manipulation of these files (when shape is updated or modes mapped onto surface)
- How does one generate this surface info?
  - Use the CLO **`--write_massoud_file`** and **`&massoud_output`** namelist input during an initial run (perhaps when generating a rigid steady-state solution)
  - This will generate a **`[project]_massoud_bodyN.(dat or ddfb)`** file for input to DDFdrive or as a template for some other middleware.
  - Rename as needed (e.g. **`[project]_bodyN`** for static AE)

http://fun3d.larc.nasa.gov FUN3D Training Workshop June 20-21, 2015 11



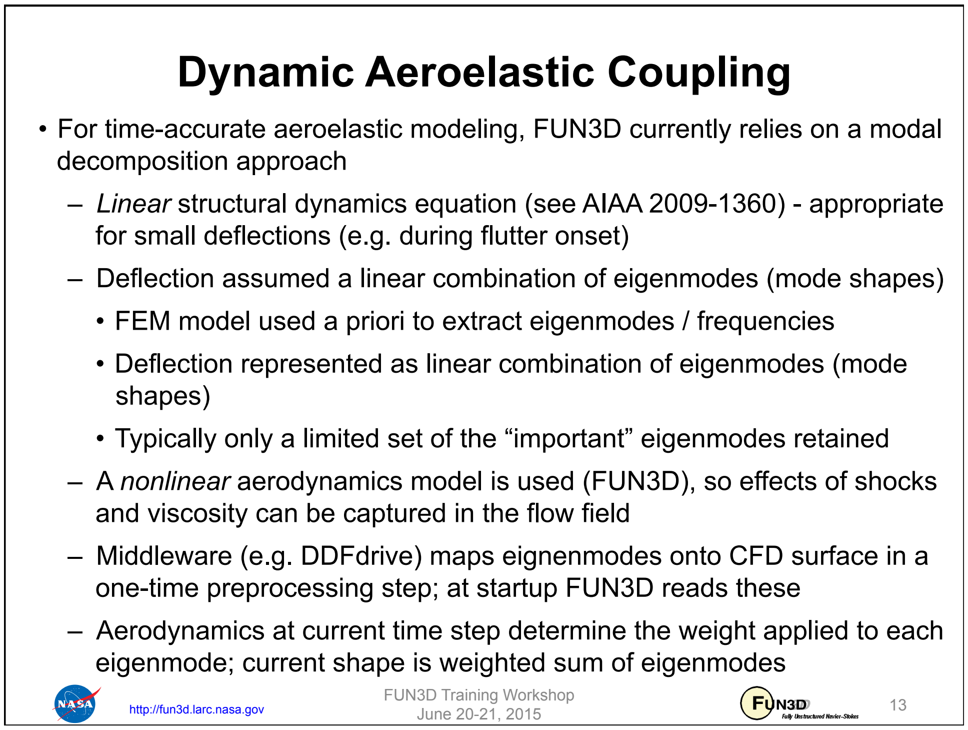
“Bootstrapping” Aeroelastic Problems (2/2)

- Example

```
&massoud_output
  n_bodies   = 2
  nbndry(1)  = 3
  boundary_list(1)  =  ‘5 7 9’
  nbndry(2)  = 2
  boundary_list(2)  =  ‘3 4’
/
```

- Also need CLO **`--write_massoud_file`**

http://fun3d.larc.nasa.gov FUN3D Training Workshop June 20-21, 2015 12

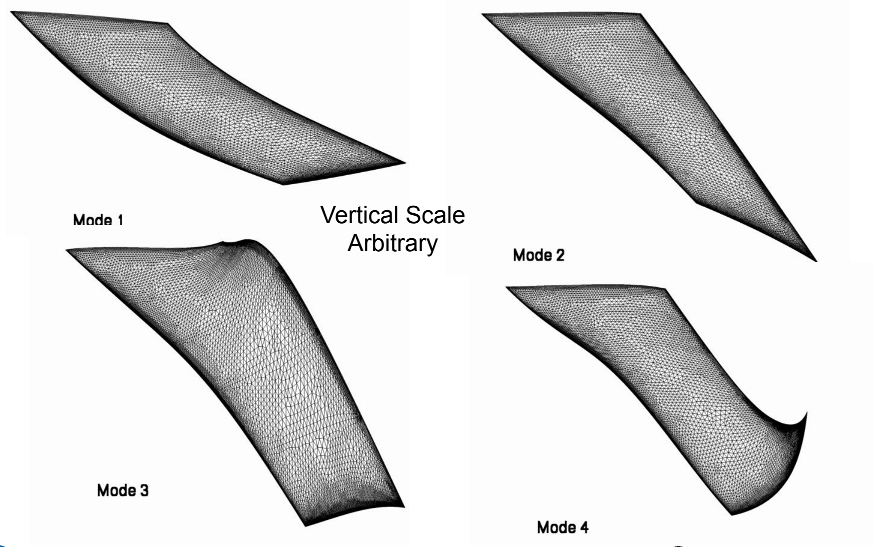


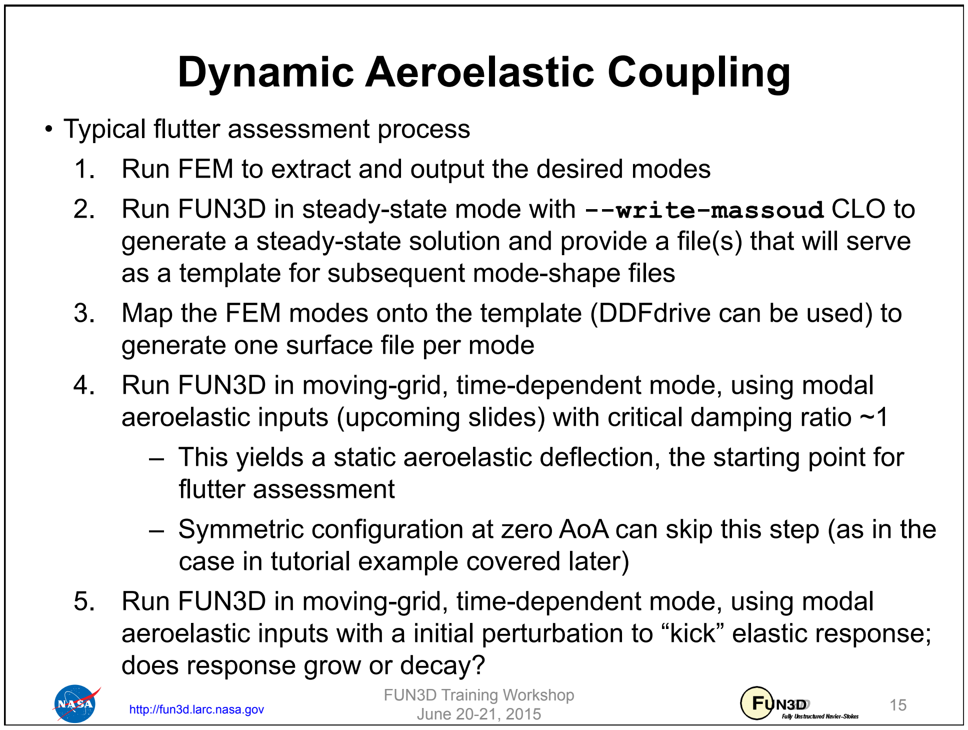
Dynamic Aeroelastic Coupling

- For time-accurate aeroelastic modeling, FUN3D currently relies on a modal decomposition approach
  - *Linear* structural dynamics equation (see AIAA 2009-1360) - appropriate for small deflections (e.g. during flutter onset)
  - Deflection assumed a linear combination of eigenmodes (mode shapes)
    - FEM model used a priori to extract eigenmodes / frequencies
    - Deflection represented as linear combination of eigenmodes (mode shapes)
    - Typically only a limited set of the “important” eigenmodes retained
  - A *nonlinear* aerodynamics model is used (FUN3D), so effects of shocks and viscosity can be captured in the flow field
  - Middleware (e.g. DDFdrive) maps eignenmodes onto CFD surface in a one-time preprocessing step; at startup FUN3D reads these
  - Aerodynamics at current time step determine the weight applied to each eigenmode; current shape is weighted sum of eigenmodes

Dynamic Aeroelastic Coupling

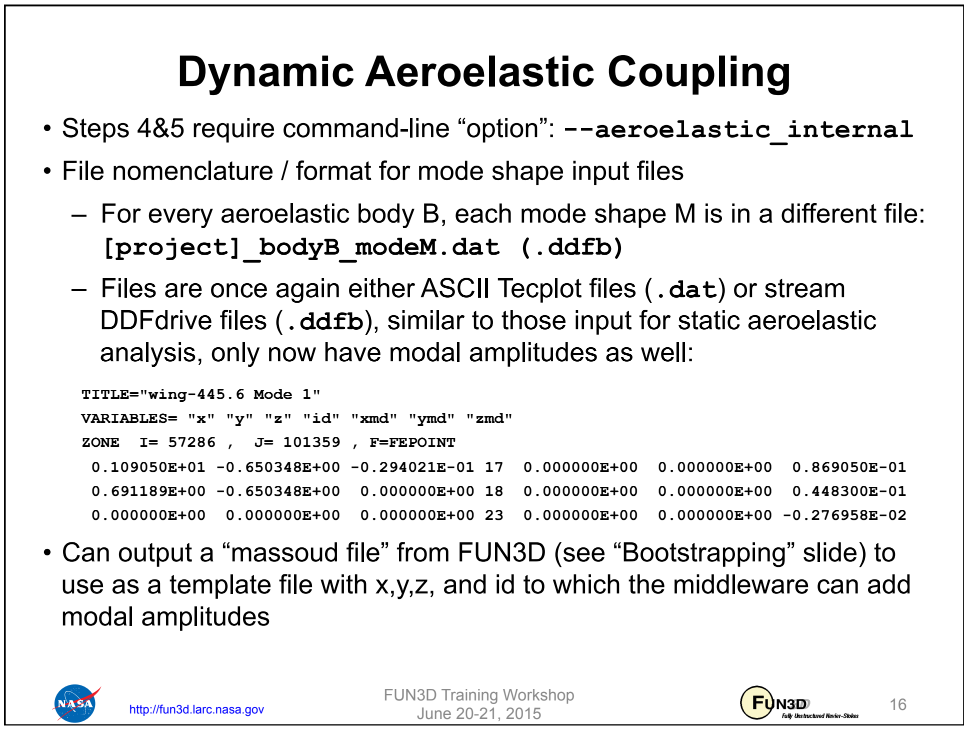
First 4 Mode Shapes AGARD 445 Wing





Dynamic Aeroelastic Coupling

- Typical flutter assessment process
  1. Run FEM to extract and output the desired modes
  2. Run FUN3D in steady-state mode with **`--write-massoud`** CLO to generate a steady-state solution and provide a file(s) that will serve as a template for subsequent mode-shape files
  3. Map the FEM modes onto the template (DDFdrive can be used) to generate one surface file per mode
  4. Run FUN3D in moving-grid, time-dependent mode, using modal aeroelastic inputs (upcoming slides) with critical damping ratio ~1
     – This yields a static aeroelastic deflection, the starting point for flutter assessment
     – Symmetric configuration at zero AoA can skip this step (as in the case in tutorial example covered later)
  5. Run FUN3D in moving-grid, time-dependent mode, using modal aeroelastic inputs with a initial perturbation to "kick" elastic response; does response grow or decay?



Dynamic Aeroelastic Coupling

- Steps 4&5 require command-line "option": **`--aeroelastic_internal`**
- File nomenclature / format for mode shape input files
  – For every aeroelastic body B, each mode shape M is in a different file: **`[project]_bodyB_modeM.dat (.ddfb)`**
  – Files are once again either ASCII Tecplot files (**`.dat`**) or stream DDFdrive files (**`.ddfb`**), similar to those input for static aeroelastic analysis, only now have modal amplitudes as well:

```
TITLE="wing-445.6 Mode 1"
VARIABLES= "x" "y" "z" "id" "xmd" "ymd" "zmd"
ZONE  I= 57286 ,  J= 101359 , F=FEPOINT
 0.109050E+01 -0.650348E+00 -0.294021E-01 17  0.000000E+00  0.000000E+00  0.869050E-01
 0.691189E+00 -0.650348E+00  0.000000E+00 18  0.000000E+00  0.000000E+00  0.448300E-01
 0.000000E+00  0.000000E+00  0.000000E+00 23  0.000000E+00  0.000000E+00 -0.276958E-02
```

- Can output a "massoud file" from FUN3D (see "Bootstrapping" slide) to use as a template file with x,y,z, and id to which the middleware can add modal amplitudes

Tutorial Case: AGARD 445 Wing (1/8)

- Test case located in: tutorials/flow_modal_aeroelasticity
  - **`run_tutorial.sh`** script performs steps 2,3, and 5 of the typical flutter assessment process
    - FEM mode shapes (Step 1) are given in Modes/445.6-mode.dat
    - No need for Step 4: symmetric airfoil at 0 deg. AoA
    - This tutorial case takes several hours to run
- Well-known test case for flutter prediction
  - Tested in NASA Langley Transonic Dynamics Tunnel c. 1960
  - Often run as inviscid (as we do here)
  - Typically run over a range of transonic Mach numbers to see at what dynamic pressure the wing begins to flutter (and with what frequency): here we consider only Mach 0.9 and q = 75 psf

Tutorial Case: AGARD 445 Wing (2/8)

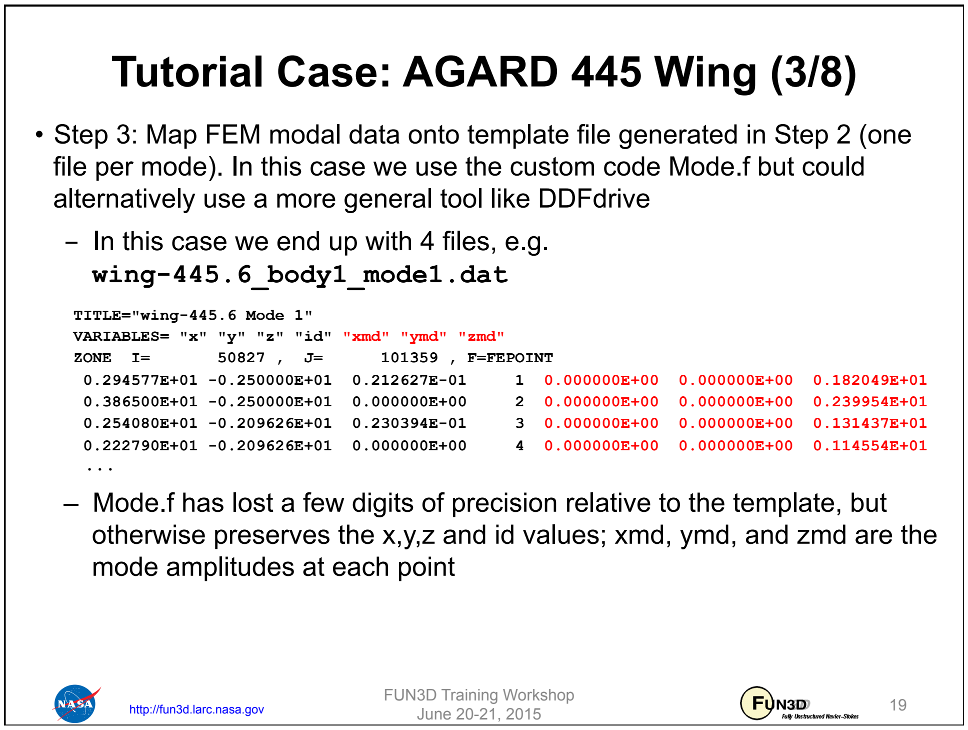
- Step 2: Generate template for FUN3D mode-shape files
  - Typical steady-state run, but with CLO **`--write_massoud_file`**, and fun3d.nml namelist input:

```
&massoud_output
  n_bodies          = 1
  nbndry(1)         = 1   ! Note:family lumping -> 1 boundary
  boundary_list(1)  = '3' ! Lumped wing is now boundary no. 3
/
```

  - Generates file wing-445.6_massoud_body1.dat:

```
title="surface points and l2g id for massoud"
variables="x","y","z","id"
zone t="mdo body 1", i=50827, j=101359, f=fepoint,  solutiontime= 0.8000000E+03,
strandid=0
   0.294576800000000E+001 -0.250000000000000E+001  0.212627299999966E-001         1
   0.386499999999999E+001 -0.250000000000000E+001  0.000000000000000E+000         2
   0.254080100000000E+001 -0.209625900000000E+001  0.230393900000010E-001         3
   0.222790400000000E+001 -0.209625900000000E+001  0.000000000000000E+000         4
   ...
```

  - FUN3D mode files must preserve these x,y,z *and* id values, and append the x,y,z modal amplitude at the end of each line – Mode.f does this



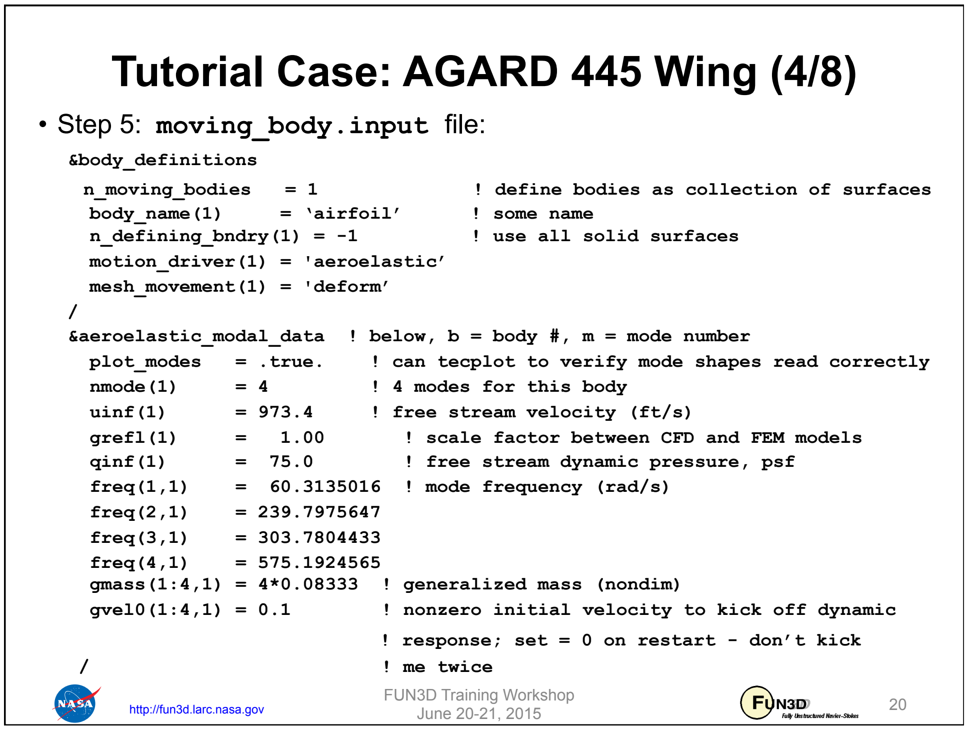
Tutorial Case: AGARD 445 Wing (3/8)

- Step 3: Map FEM modal data onto template file generated in Step 2 (one file per mode). In this case we use the custom code Mode.f but could alternatively use a more general tool like DDFdrive
  - In this case we end up with 4 files, e.g. **`wing-445.6_body1_mode1.dat`**

```
TITLE="wing-445.6 Mode 1"
VARIABLES= "x" "y" "z" "id" "xmd" "ymd" "zmd"
ZONE  I=       50827 ,  J=      101359 , F=FEPOINT
 0.294577E+01 -0.250000E+01  0.212627E-01     1  0.000000E+00  0.000000E+00  0.182049E+01
 0.386500E+01 -0.250000E+01  0.000000E+00     2  0.000000E+00  0.000000E+00  0.239954E+01
 0.254080E+01 -0.209626E+01  0.230394E-01     3  0.000000E+00  0.000000E+00  0.131437E+01
 0.222790E+01 -0.209626E+01  0.000000E+00     4  0.000000E+00  0.000000E+00  0.114554E+01
 ...
```

  - Mode.f has lost a few digits of precision relative to the template, but otherwise preserves the x,y,z and id values; xmd, ymd, and zmd are the mode amplitudes at each point

http://fun3d.larc.nasa.gov FUN3D Training Workshop June 20-21, 2015 FUN3D Fully Unstructured Navier-Stokes 19

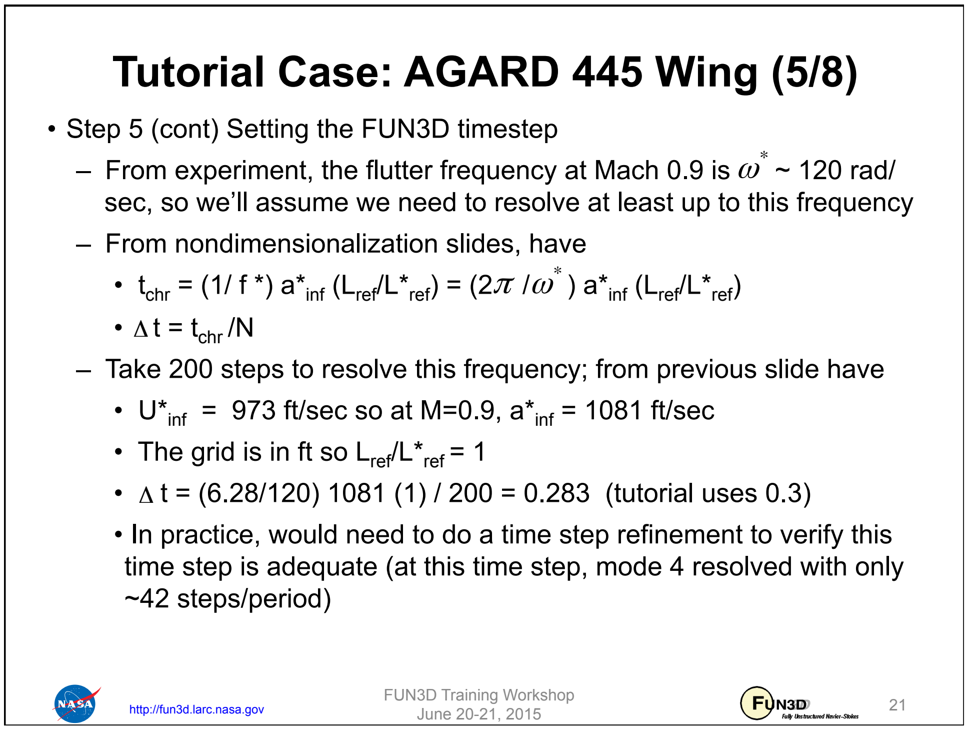


Tutorial Case: AGARD 445 Wing (4/8)

- Step 5: **`moving_body.input`** file:

```
&body_definitions
  n_moving_bodies    = 1              ! define bodies as collection of surfaces
   body_name(1)      = 'airfoil'      ! some name
   n_defining_bndry(1) = -1           ! use all solid surfaces
   motion_driver(1) = 'aeroelastic'
   mesh_movement(1) = 'deform'
/
&aeroelastic_modal_data  ! below, b = body #, m = mode number
   plot_modes    = .true.    ! can tecplot to verify mode shapes read correctly
   nmode(1)      = 4         ! 4 modes for this body
   uinf(1)       = 973.4     ! free stream velocity (ft/s)
   grefl(1)      =   1.00       ! scale factor between CFD and FEM models
   qinf(1)       =  75.0        ! free stream dynamic pressure, psf
   freq(1,1)     =  60.3135016  ! mode frequency (rad/s)
   freq(2,1)     = 239.7975647
   freq(3,1)     = 303.7804433
   freq(4,1)     = 575.1924565
   gmass(1:4,1) = 4*0.08333   ! generalized mass (nondim)
   gvel0(1:4,1) = 0.1         ! nonzero initial velocity to kick off dynamic
                              ! response; set = 0 on restart - don't kick
 /                            ! me twice
```

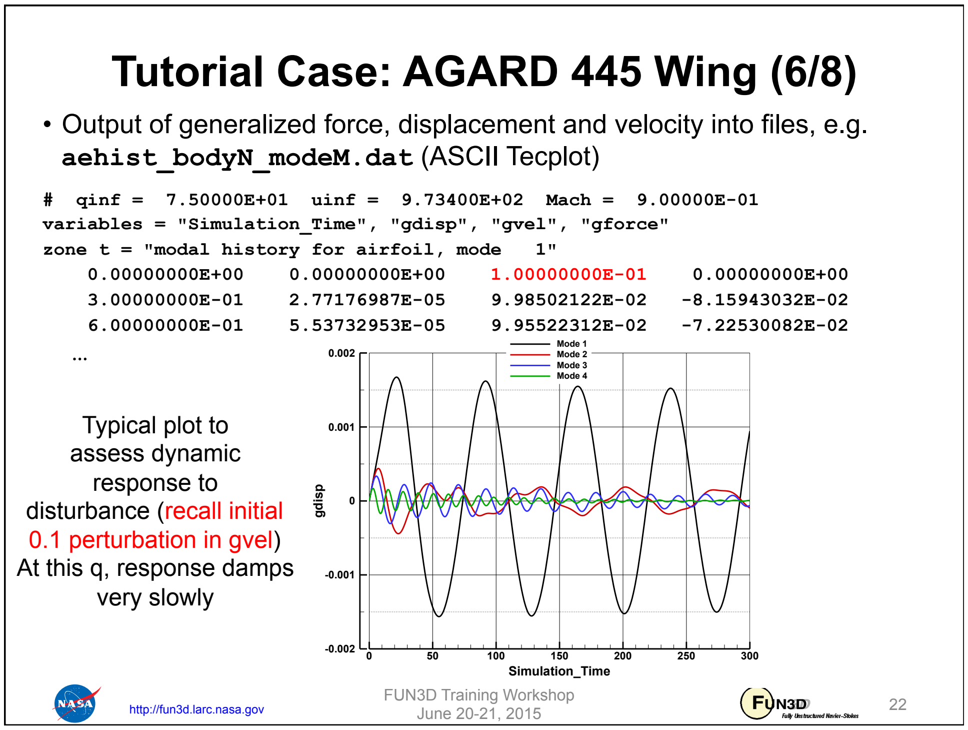
http://fun3d.larc.nasa.gov FUN3D Training Workshop June 20-21, 2015 FUN3D Fully Unstructured Navier-Stokes 20



Tutorial Case: AGARD 445 Wing (5/8)

- Step 5 (cont) Setting the FUN3D timestep
  - From experiment, the flutter frequency at Mach 0.9 is ω^* ~ 120 rad/sec, so we'll assume we need to resolve at least up to this frequency
  - From nondimensionalization slides, have
    - $t_{chr} = (1/f^*)\, a^*_{inf}\, (L_{ref}/L^*_{ref}) = (2\pi/\omega^*)\, a^*_{inf}\, (L_{ref}/L^*_{ref})$
    - $\Delta t = t_{chr}/N$
  - Take 200 steps to resolve this frequency; from previous slide have
    - U^*_{inf} = 973 ft/sec so at M=0.9, a^*_{inf} = 1081 ft/sec
    - The grid is in ft so $L_{ref}/L^*_{ref} = 1$
    - Δt = (6.28/120) 1081 (1) / 200 = 0.283 (tutorial uses 0.3)
    - In practice, would need to do a time step refinement to verify this time step is adequate (at this time step, mode 4 resolved with only ~42 steps/period)

http://fun3d.larc.nasa.gov FUN3D Training Workshop June 20-21, 2015 FUN3D Fully Unstructured Navier-Stokes 21

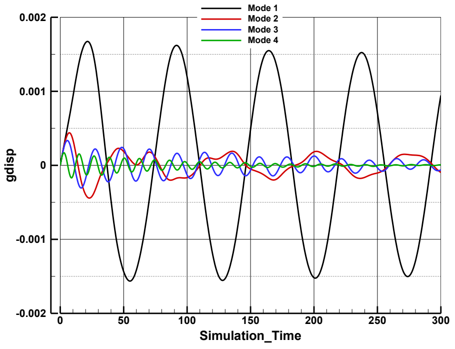


Tutorial Case: AGARD 445 Wing (6/8)

- Output of generalized force, displacement and velocity into files, e.g. **`aehist_bodyN_modeM.dat`** (ASCII Tecplot)

```
#  qinf =   7.50000E+01  uinf =   9.73400E+02  Mach =   9.00000E-01
variables = "Simulation_Time", "gdisp", "gvel", "gforce"
zone t = "modal history for airfoil, mode   1"
     0.00000000E+00    0.00000000E+00    1.00000000E-01    0.00000000E+00
     3.00000000E-01    2.77176987E-05    9.98502122E-02   -8.15943032E-02
     6.00000000E-01    5.53732953E-05    9.95522312E-02   -7.22530082E-02
   …
```

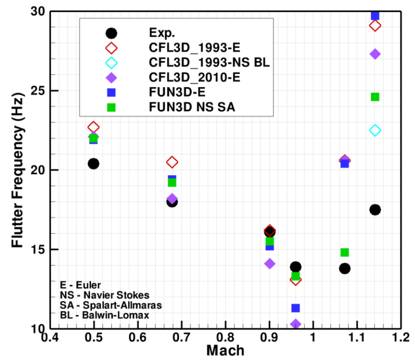
Typical plot to assess dynamic response to disturbance (recall initial 0.1 perturbation in gvel)
At this q, response damps very slowly

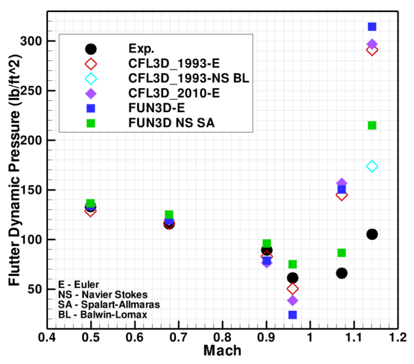


http://fun3d.larc.nasa.gov FUN3D Training Workshop June 20-21, 2015 FUN3D Fully Unstructured Navier-Stokes 22

Tutorial Case: AGARD 445 Wing (7/8)

- The dynamic pressure (q = 75 psf) in the tutorial does not lead to flutter at M = 0.9 – so we would need to increment q and repeat until we found a response that grows with time (M = 78.6 psf) – then repeat over the Mach range
- Pawel Chwalowski, Aeroelasticity Branch, NASA Langley has carried out this exercise and provided these plots (not part of tutorial):



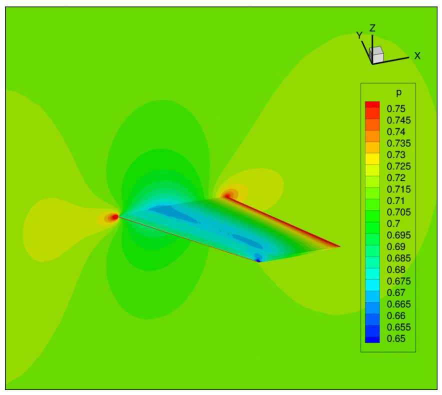


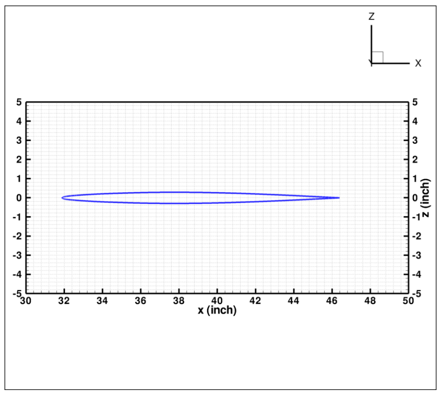
http://fun3d.larc.nasa.gov FUN3D Training Workshop June 20-21, 2015 FUN3D 23

Tutorial Case: AGARD 445 Wing (8/8)

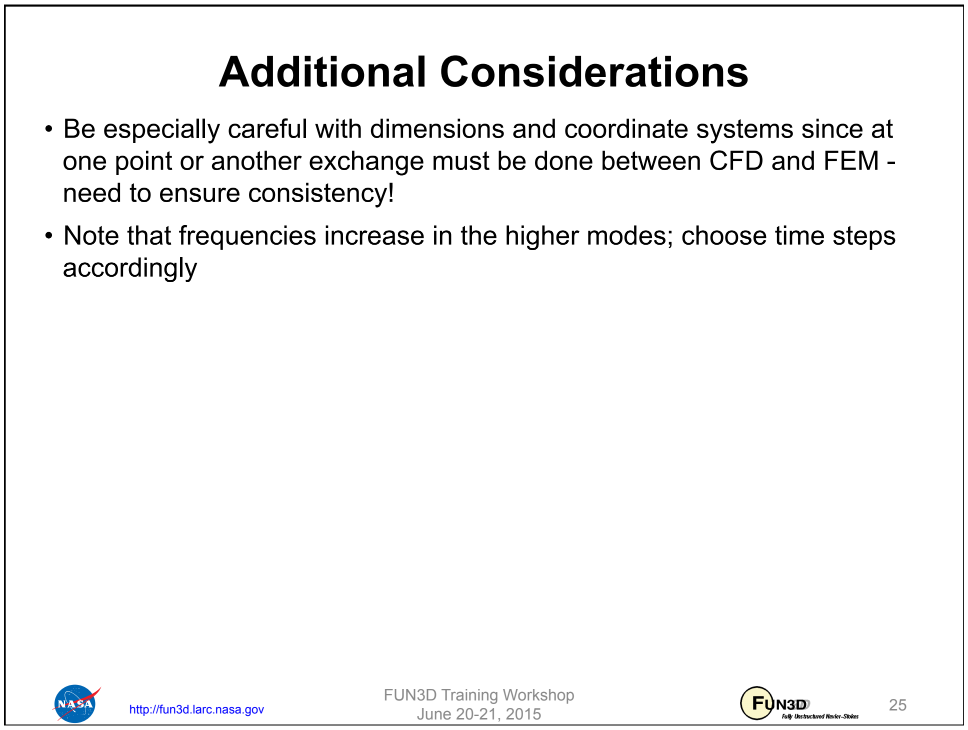
Results Courtesy Pawel Chwalowski, Aeroelasticity Branch, NASA Langley
(These animations not generated as part of the tutorial)

Inviscid Flow Mach=0.9, Flutter condition, Q = 78.6 psf





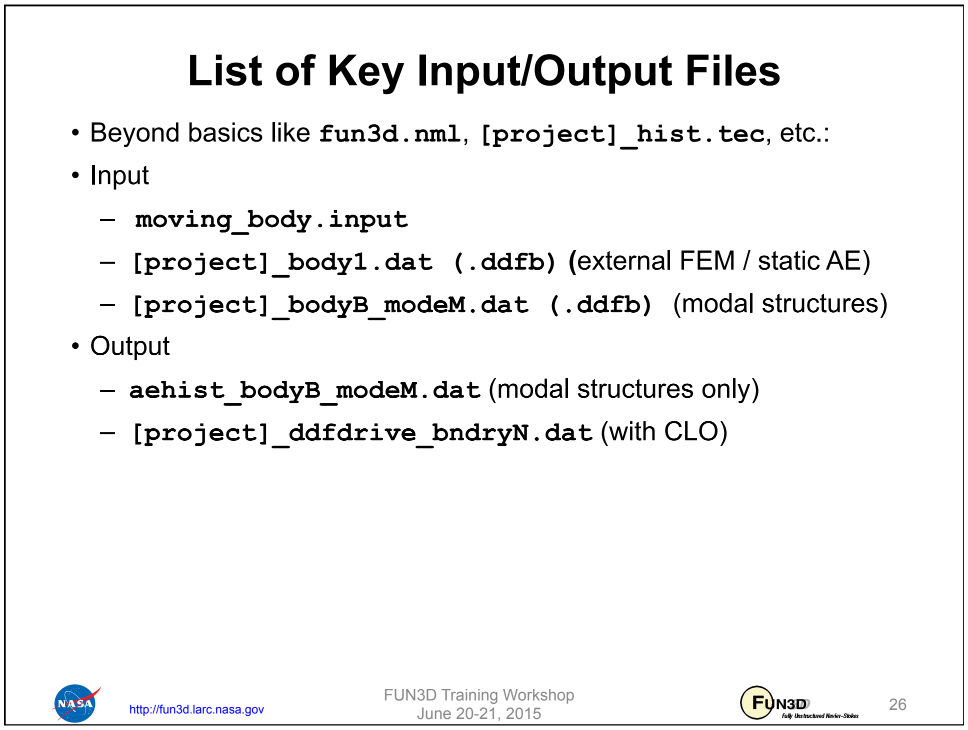
http://fun3d.larc.nasa.gov FUN3D Training Workshop June 20-21, 2015 FUN3D 24



Additional Considerations

- Be especially careful with dimensions and coordinate systems since at one point or another exchange must be done between CFD and FEM - need to ensure consistency!
- Note that frequencies increase in the higher modes; choose time steps accordingly

http://fun3d.larc.nasa.gov FUN3D Training Workshop June 20-21, 2015 25



List of Key Input/Output Files

- Beyond basics like **`fun3d.nml`**, **`[project]_hist.tec`**, etc.:
- Input
  - **`moving_body.input`**
  - **`[project]_body1.dat (.ddfb)`** (external FEM / static AE)
  - **`[project]_bodyB_modeM.dat (.ddfb)`** (modal structures)
- Output
  - **`aehist_bodyB_modeM.dat`** (modal structures only)
  - **`[project]_ddfdrive_bndryN.dat`** (with CLO)

http://fun3d.larc.nasa.gov FUN3D Training Workshop June 20-21, 2015 26