

FUN3D v12.7 Training

Session 11: Time-Dependent Simulations

Bob Biedron



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

FUN3D Training Workshop
June 20-21, 2015



1

Session Scope

- What this will cover
 - How to set up and run time-accurate simulations on static meshes
 - Subiteration convergence: what to strive for and why
 - Nondimensionalization
 - Choosing the time step
 - Input / Output
- What will not be covered
 - Moving-mesh, aeroelastics (covered in follow-on sessions)
- What should you already be familiar with
 - Basic steady-state solver operation and control
 - Basic flow visualization



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

FUN3D Training Workshop
June 20-21, 2015



2

Introduction

- Background
 - Many of problems of interest involve unsteady flows and may also involve moving geometries
 - Governing equations written in Arbitrary Lagrangian-Eulerian (ALE) form to account for grid speed
 - Nondimensionalization often more involved/confusing/critical
- Compatibility
 - Compressible/incompressible paths
 - Mixed elements; 2D/3D
 - Dynamic grids
 - Not compatible with generic gas model
- Status
 - Incompressible path exercised very infrequently for unsteady flows



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

FUN3D Training Workshop
June 20-21, 2015



3

Governing Equations

- Arbitrary Lagrangian-Eulerian (ALE) Formulation

$$\frac{\partial(\bar{Q}V)}{\partial t} = -\oint_{\partial V} (\bar{F} - \bar{q}\bar{W}^T) \cdot \bar{n} dS - \oint_{\partial V} \bar{F}_v \cdot \bar{n} dS = \bar{R} \quad \bar{Q} = \frac{\oint_V \bar{q} dV}{V}$$

\bar{W} = Arbitrary control surface velocity; Lagrangian if $\bar{W} = (u, v, w)^T$ (moves with fluid); Eulerian if $\bar{W} = 0$ (fixed in space)

- Discretize using Nth order backward differences in time, linearize \bar{R} about time level n+1, and introduce a pseudo-time term:

$$\left[\left(\frac{V^{n+1}}{\Delta\tau} + \frac{V^{n+1}\phi_{n+1}}{\Delta t} \right) \bar{I} - \frac{\partial \bar{R}^{n+1,m}}{\partial \bar{Q}} \right] \Delta \bar{Q}^{n+1,m} = \bar{R}^{n+1,m} - \frac{V^{n+1}\phi_{n+1}}{\Delta t} (\bar{Q}^{n+1,m} - \bar{Q}^n) - \dots + \bar{R}_{GCL}^{n+1} = \underline{\bar{R}^{n+1,m}} + O(\Delta t^N)$$

- Physical time-level t^n ; Pseudo-time level τ^m
- Want to drive **subiteration residual** $\bar{R}^{n+1,m} \rightarrow 0$ using pseudo-time subiterations at each time step – more later – otherwise you have more error than the expected $O(\Delta t^N)$ truncation error



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

FUN3D Training Workshop
June 20-21, 2015



4

Time Advancement - Namelist Input

- The `&nonlinear_solver_parameters` namelist in the `fun3d.nml` file governs how the solution is advanced in time
- Relevant entries - *default values shown* - some definitely need changing:

```
&nonlinear_solver_parameters
  time_accuracy      = 'steady' (i.e. not time accurate)
  time_step_nondim   = 0.0
  subiterations      = 0
  schedule_iteration = 1      50
  schedule_cfl       = 200.0 200.0
  schedule_cfl_turb  = 50.0  50.0
  pseudo_time_stepping = "on"
  temporal_err_control = .false.
  temporal_err_floor  = 0.1
/
```

- Let's look at these in some detail (defer `time_step_nondim` to last)



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

FUN3D Training Workshop
June 20-21, 2015



5

Time Advancement - Order of Accuracy

- Currently have several types of backward difference formulae (BDF) that are controlled by the `time_accuracy` component:
 - In order of formal accuracy: BDF1 (**1storder**), BDF2 (**2ndorder**), BDF2_{OPT} (**2ndorderOPT**), BDF3 (**3rdorder**), MEBDF4 (**4thorderMEBDF4**)
 - Can pretty much ignore all but BDF2_{OPT} and BDF2
 - BDF1 is least accurate; little gain in CPU time / step over 2nd order; for moving grids can be helpful to start out with BDF1 (rare)
 - BDF3 not guaranteed to be stable; feeling lucky?
 - MEBDF4 only efficient if working to very high levels of accuracy - *including spatial accuracy* - generally **not** for practical problems
 - BDF2_{OPT} (*recommended*) is a stable blend of BDF2 and BDF3 schemes; formally 2nd order accurate but error is ~1/2 that of BDF2; also allows for a more accurate estimate of the temporal error for the error controller (p.8)



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

FUN3D Training Workshop
June 20-21, 2015



6

Time Advancement - Subiterations (1/4)

- Can think of each time step as a mini steady-state problem
- Subiterations (`subiterations > 0`) are essential
 - Subiteration control in *each time step* operates exactly like iteration control in a steady state case:
 - CFL ramping is available for mean flow and turbulence model – however, be aware that ramping schedule should be `< subiterations` or the specified final CFL won't be obtained
 - *We almost never ramp CFL for time-accurate cases*
 - If used, CFL ramping starts over each time step
 - Caution: the *spatial* accuracy flag, `first_order_iterations`, starts over each time step, so make sure you don't have this on
- Pseudo-time term helpful for large time steps
 - We *always* use it in our applications
 - `pseudo_time_stepping = "on"` (default)



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

FUN3D Training Workshop
June 20-21, 2015



7

Time Advancement - Subiterations (2/4)

- How many subiterations?
 - In theory, should drive subiteration residual “to zero” each time step – but you cannot afford to do that
 - Otherwise have additional errors other than $O(\Delta t^2)$ (if 2nd order time)
- In a perfect world, the answer is to use the **temporal error controller**
 - `temporal_err_control = .true.`
 - `temporal_err_floor = 0.1` => iterate until the subiteration residual is 1 order lower than the (estimated) temporal error (0.01 => 2)
 - Subiterations kick out when this level of convergence is reached OR subiteration counter `> subiterations`
 - (empirically) 1 order is about the minimum; 2 orders is better, BUT...
 - Often, either the turbulence residual converges slowly or the mean flow does, and the max subiterations you specify will be reached
 - When it kicks in, the temporal error controller is the best approach, and the most efficient; even if it doesn't kick in, it can be informative



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

FUN3D Training Workshop
June 20-21, 2015



8

Time Advancement - Subiterations (3/4)

- Be wary reaching conclusions about the effect of time-step refinement unless the subiterations are “sufficiently” converged for each size step
- How to monitor and assess the subiteration convergence:
 - Printed to the screen, so you can “eyeball” it
 - With temporal error controller, if the requested tolerance is not met, message(s) will be output to the screen:
 - **WARNING: mean flow subiterations failed to converge to specified temporal_err_floor level**
 - **WARNING: turb flow subiterations failed to converge to specified temporal_err_floor level**
 - Note: when starting unsteady mode, first timestep *never* achieves target error (no error estimate first step, so target is 0)
 - Note: x-momentum residual (R_2) is the mean-flow residual targeted by the error controller
- Plot it (usually best)



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

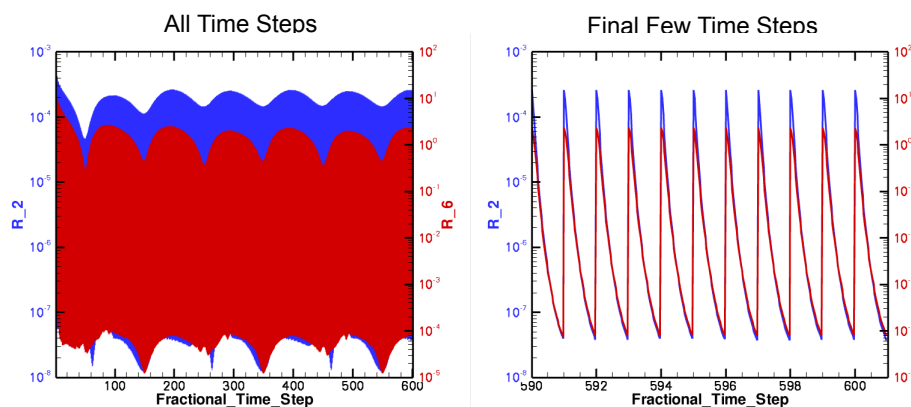
FUN3D Training Workshop
June 20-21, 2015



9

Time Advancement - Subiterations (4/4)

- Tecplot file (ASCII) with subiteration convergence history is output to a file: `[project]_subhist.dat`
 - Plot (on log scale) R_2 (etc) vs `Fractional_Time_Step`
 - Also contains C_l , C_d , C_m to assess force convergence in a time step



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

FUN3D Training Workshop
June 20-21, 2015



10

Nondimensionalization of Time

- Notation: * indicates a dimensional variable, otherwise nondimensional; the reference flow state is usually free stream ("∞"), but need not be
- Define:
 - L_{ref}^* = reference length of the physical problem (e.g. chord in ft)
 - L_{ref} = corresponding length in your grid (considered *nondimensional*)
 - a_{ref}^* = reference speed of sound (e.g. ft/sec) (compressible)
 - U_{ref}^* = reference velocity (e.g. ft/sec; compressible: $U_{ref}^* = \text{Mach } a_{ref}^*$)
 - t^* = time (e.g. sec)
- Then nondimensional time in FUN3D is related to physical time by:
 - $t = t^* a_{ref}^* (L_{ref}/L_{ref}^*)$ (compressible)
 - $t = t^* U_{ref}^* (L_{ref}/L_{ref}^*)$ (incompressible)
 - *Usually* have $L_{ref}/L_{ref}^* = 1^*$, but need not - e.g. typical 2D airfoil grid
 - L_{ref}/L_{ref}^* appears because Re in FUN3D is input *per unit grid length*



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

FUN3D Training Workshop
June 20-21, 2015



11

Determining the Time Step

- Identify a **characteristic time** t_{chr}^* that you need to resolve with some level of accuracy in your simulation; perhaps:
 - Some important shedding frequency f_{shed}^* (Hz) is known or estimated
 $t_{chr}^* \sim 1 / f_{shed}^*$
 - Periodic motion of the body $t_{chr}^* \sim 1 / f_{motion}^*$
 - A range of frequencies in a DES-type simulation $t_{chr}^* \sim 1 / f_{highest}^*$
 - If none of the above, you can estimate the time it takes for a fluid particle to cross the characteristic length of the body, $t_{chr}^* \sim L_{ref}^* / U_{ref}^*$
 - $t_{chr} = t_{chr}^* a_{ref}^* (L_{ref}/L_{ref}^*)$ (comp) $t_{chr} = t_{chr}^* U_{ref}^* (L_{ref}/L_{ref}^*)$ (incomp)
- Say you want N time steps within the characteristic time:
 - $\Delta t = t_{chr} / N = \text{time_step_nondim}$
- Figure an absolute *minimum* of $N = 100$ for reasonable resolution of t_{chr} with a 2nd order scheme - really problem dependent (*frequencies* $> f^*$ *may be important*); but don't over resolve time if space is not well resolved too



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

FUN3D Training Workshop
June 20-21, 2015



12

Tutorial Case: Unsteady Flow, High AoA (1/7)

- Test case located in: tutorials/flow_unsteady_airfoil_high_AoA
 - `run_tutorial.sh` script starts with a 2000 time step restart file, runs an additional 100 steps, and makes plots that follow
- Consider flow past a (2D) NACA 0012 airfoil at 45° angle of attack - the flow separates and is unsteady
 - $Re_{c^*} = 4.8$ million, $M_{ref} = 0.6$, assume $a^*_{ref} = 340$ m/s
 - chord = 0.1m, chord-in-grid = 1.0 so $L_{ref}/L^*_{ref} = 1.0/0.1 = 10$ (m^{-1})
 - Say we know from experiment that lift oscillations occur at ~450 Hz
 - $t^*_{chr} = 1 / f^*_{chr} = 1 / 450$ Hz = 0.002222 s
 - $t_{chr} = t^*_{chr} a^*_{ref} (L_{ref}/L^*_{ref}) = (0.002222)(340)(10) = 7.555$
 - $\Delta t = t_{chr} / N$ so $\Delta t = 0.07555$ for 100 steps / lift cycle
 - By way of comparison, for $M = 0.6$, $a^*_{ref} = 340$ m/s, and $L^*_{ref} = 0.1$ m it takes a fluid particle $\sim (0.1)/(204) = 0.00049$ s to pass by the airfoil; this leads to smaller, more conservative estimate for the time step, by about a factor of 4



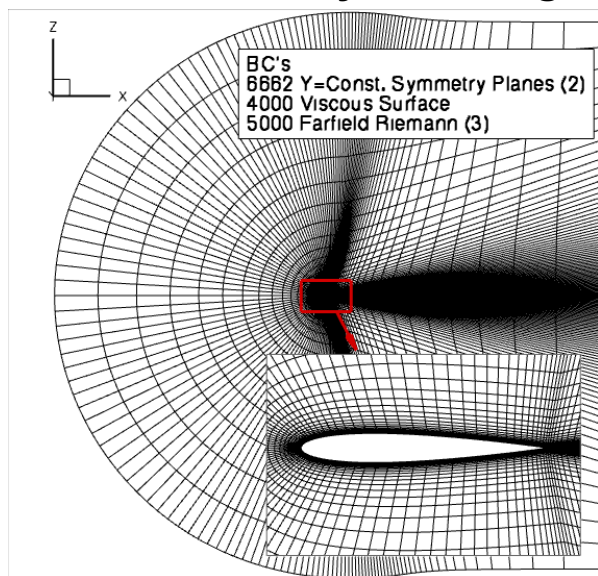
<http://fun3d.larc.nasa.gov>

FUN3D Training Workshop
June 20-21, 2015



13

Tutorial Case: Unsteady Flow, High AoA (2/7)



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

FUN3D Training Workshop
June 20-21, 2015



14

Tutorial Case: Unsteady Flow, High AoA (3/7)

- Flow viz: output u-velocity and y-component of vorticity
- Relevant fun3d.nml namelist data (note: many defaults assumed)

```

&project
  project_rootname = "n0012_i153"
  case_title = "NACA 0012 airfoil, 2D Hex Mesh"
/
&global
  boundary_animation_freq = 5
/
&raw_grid
  grid_format = "aflr3"
  data_format = "ASCII"
  twod_mode = .true.
/
&reference_physical_properties
  mach_number = 0.60
  reynolds_number = 4800000.00
  temperature = 520.00
  temperature_units = 'Rankine'
  angle_of_attack = 45.0
/

```



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

FUN3D Training Workshop
June 20-21, 2015



15

Tutorial Case: Unsteady Flow, High AoA (4/7)

- Relevant fun3d.nml namelist data (cont)

```

&force_moment_integ_properties
  x_moment_center = 0.25
/
&nonlinear_solver_parameters
  time_accuracy = "2ndorderOPT" ! Our Workhorse Scheme
  time_step_nondim = 0.07555 ! 100 steps/cycle @ 450 Hz
  temporal_err_control = .true. ! Enable error-based kickoff
  temporal_err_floor = 0.1 ! Exit 1 order below error estimate
  subiterations = 30 ! No more than 30
  schedule_cfl = 50.00 50.00 ! constant cfl each step; no ramping
  schedule_cfl_turb = 30.00 30.00
/
&code_run_control
  steps = 100 ! need ~2000 steps to be periodic from freestream
/
&boundary_output_variables
  primitive_variables = .false. ! turn off default
  y = .false. ! So tecplot displays correct 2D orientation by default
  u = .true.
  vort_y = .true.
/

```



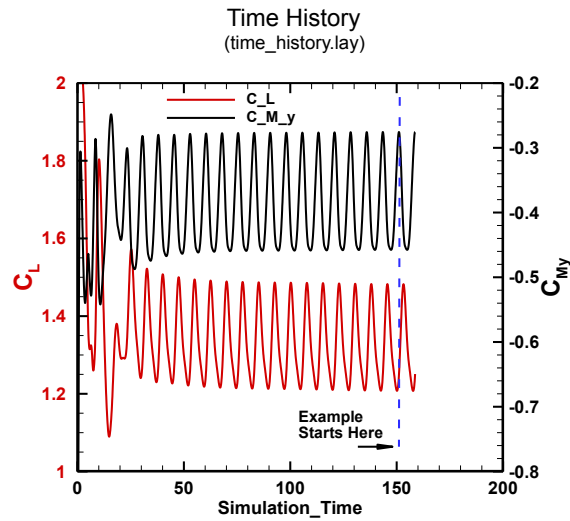
<http://fun3d.larc.nasa.gov>

FUN3D Training Workshop
June 20-21, 2015



16

Tutorial Case: Unsteady Flow, High AoA (5/7)



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

FUN3D Training Workshop
June 20-21, 2015



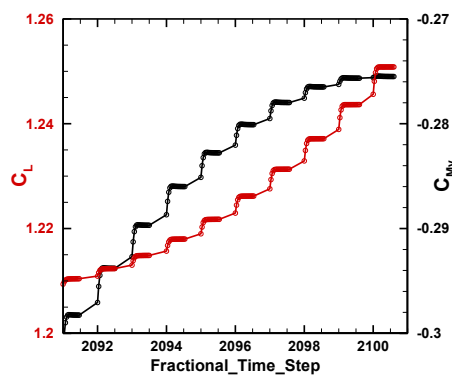
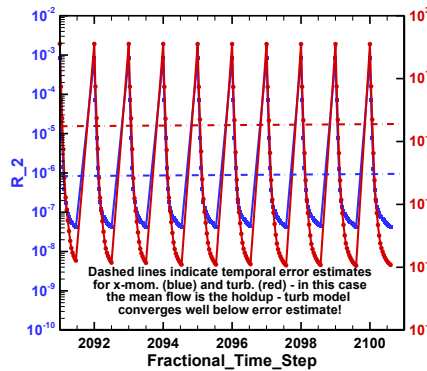
17

Tutorial Case: Unsteady Flow, High AoA (6/7)

- Subiterations converge? `grep "WARNING" screen_output | wc`
 - In this case, all steps converge to the specified tolerance

Subiteration Residuals, Final 10 Steps

Subiteration Lift & PM, Final 10 Steps



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

FUN3D Training Workshop
June 20-21, 2015

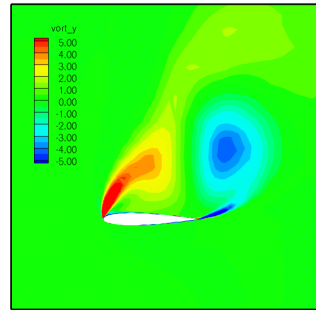
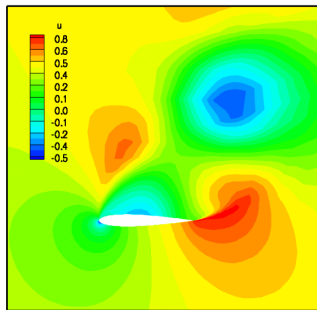


18

Tutorial Case: Unsteady Flow, High AoA (7/7)

- Animation of Results

X-Component of Velocity



Y-Component of Vorticity



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

FUN3D Training Workshop
June 20-21, 2015



19

List of Key Input/Output Files

- Beyond basics like `fun3d.nml`, etc.:
- Input
 - none
- Output
 - `[project]_subhist.dat`
 - Use to check subiteration residual and force/moment convergence



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

FUN3D Training Workshop
June 20-21, 2015



20