Missile External Aerodynamics Using Star-CCM+

Star European Conference – 03/22-23/2011

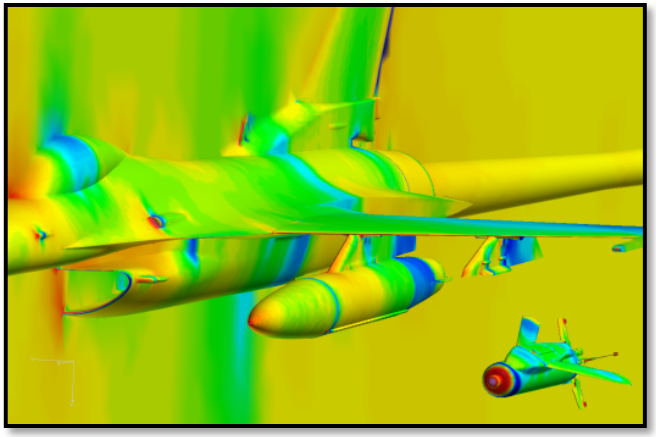
Deryl Snyder, Ph.D.
Engineering Support Group Lead
Aerodynamics Center of Excellence

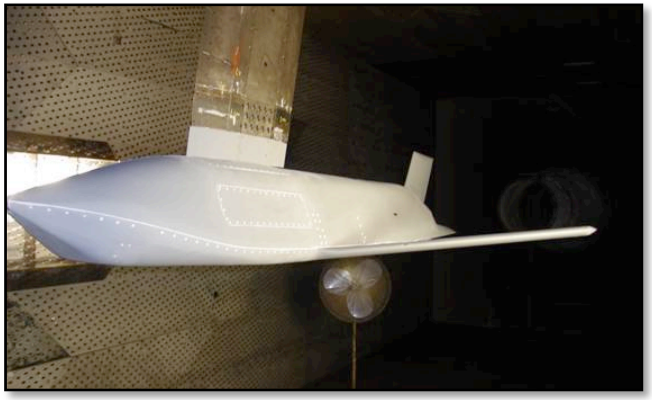
Overview

- How CFD (and, in particular, Star-CCM+) fits into the aerodynamics analysis process at Lockheed Martin Missiles and Fire Control – Orlando.
- Aerodynamic Performance Prediction Case
  - Solvers
  - Setup
  - Solution/Post-Processing Automation
  - Performance Results
- Mesh Type and Turbulence Model Selection
- Convergence Acceleration for Compressible Flows
- Conclusion

Role of CFD in Aerodynamic Analyses

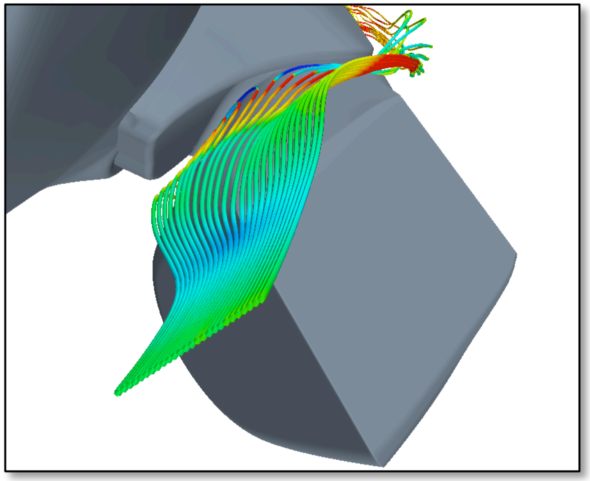
- Classical aerodynamics / Semi-Empirical
  - Bound the problem
  - Determine feasibility
  - Perform initial trades
- CFD
  - Higher fidelity performance estimation
  - Down-select to small set of geometries for WT testing
  - Determine expected WT loads
  - Identify possible trouble areas
  - Provide detailed flow information
- Wind tunnel tests
  - Final down-select
  - Final aerodynamic database

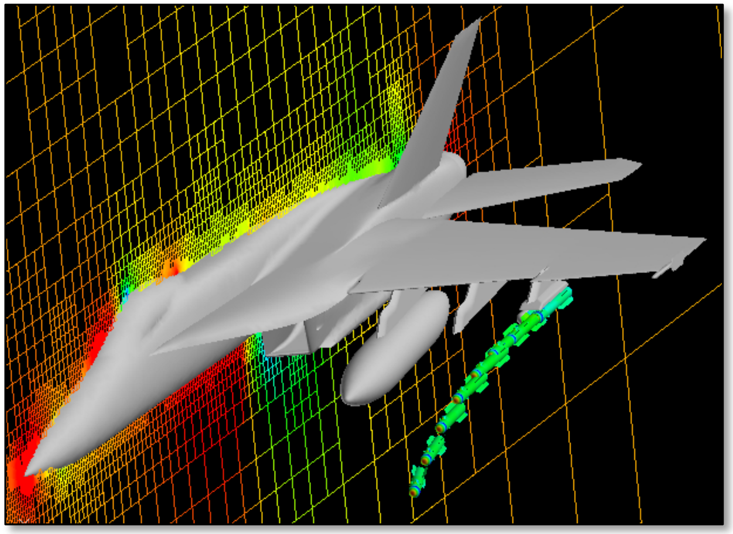




Typical CFD Applications

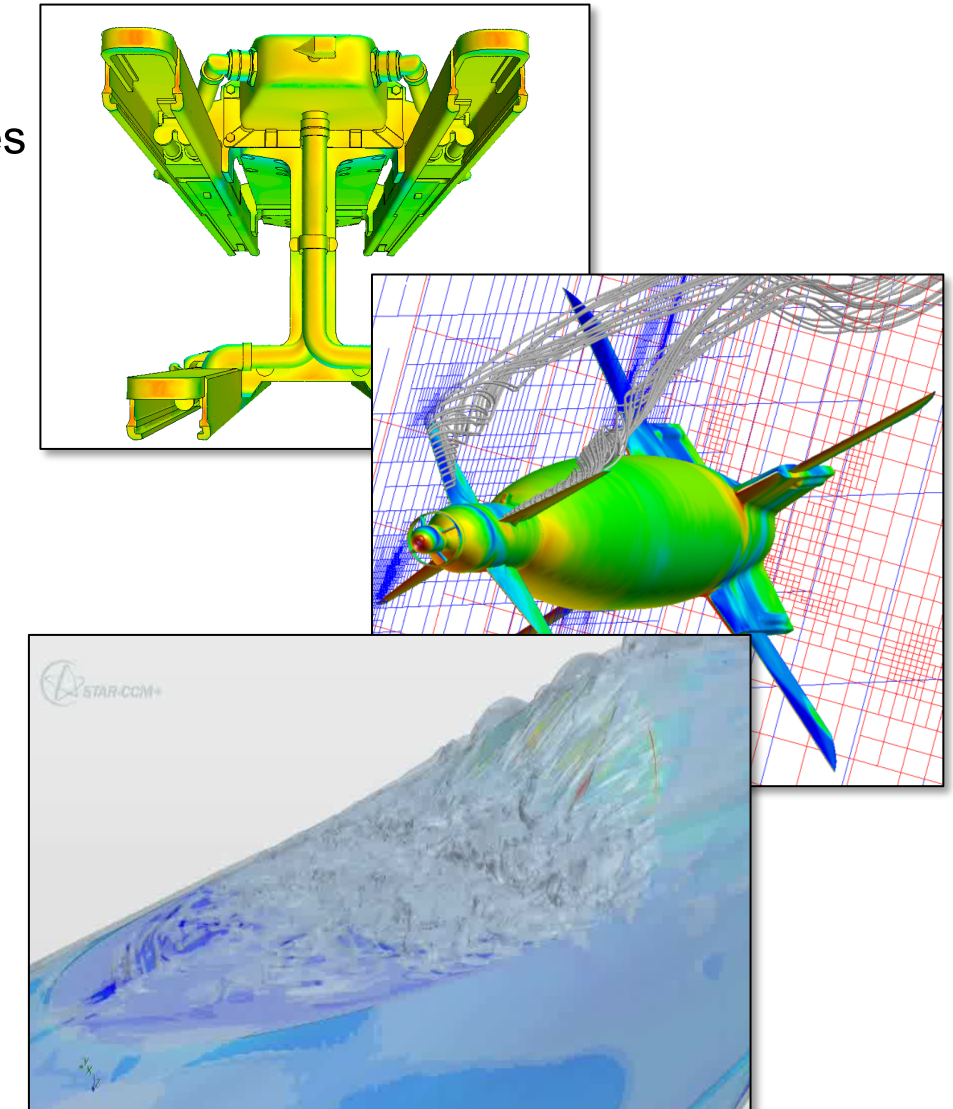
- Freestream aerodynamics
  - Estimate free-flight forces and moments
  - Generate databases for simulations
  - Identify component loading
  - Determine distributed loading for structural analysis
  - Quantify control effectiveness
- Flowfield investigations
  - Component interaction
  - Shock formation
  - Vortex interactions
  - Thermal analyses (CHT)
  - Aero-Optics
- Separation analyses
  - Estimate interference effects
  - 'Grid' approach
  - 'CFD-in-the-loop' 6-DOF simulations





Aerodynamic Demands/Trends

- Increasingly complex geometries
  - Difficult to apply classical analyses
- Increasingly complex flow fields
  - Separated flows
  - Plume interactions
  - High Mach numbers
- Increasingly difficult questions
  - Vortex interactions
  - Shock interactions
  - Optics through turbulence
  - Multiple bodies

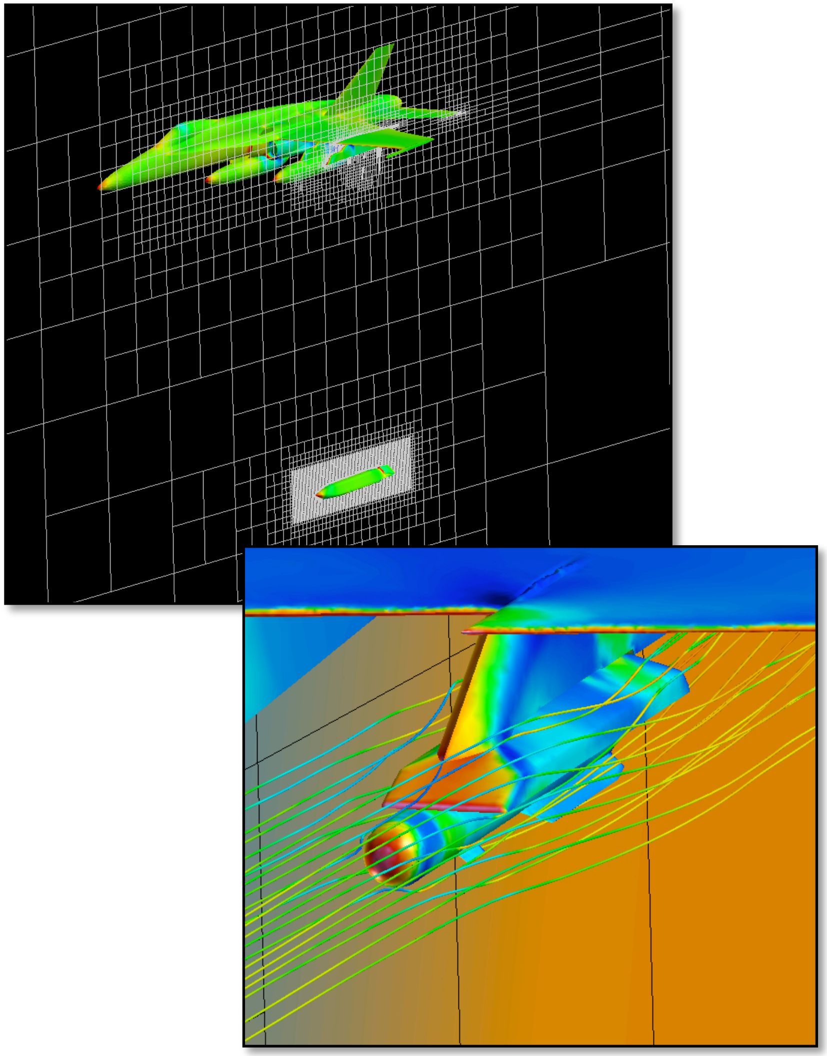


Joint Common Missile Test Case

- Joint Common Missile (JCM)
  - Freestream lift, drag, and pitching moment prediction
  - Evaluated against wind tunnel data
    - Mach: 0.5, 0.85, 1.3
    - Angle of Attack: -5 to +25 degrees
    - Sideslip Angle: 0

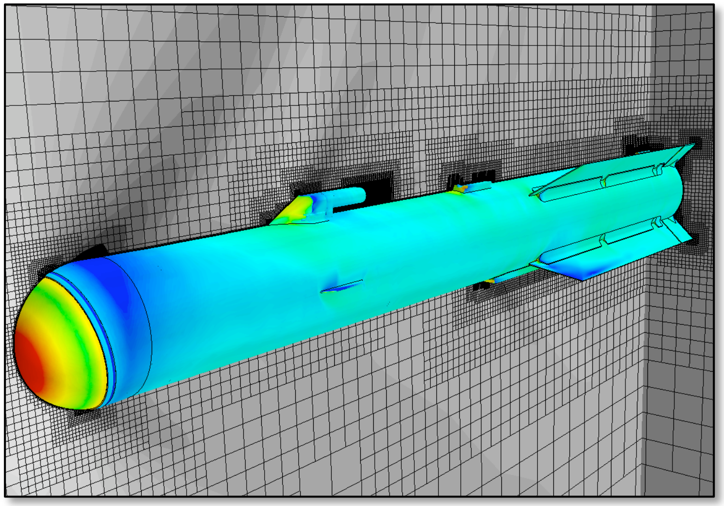
Solvers – Splitflow (LM)

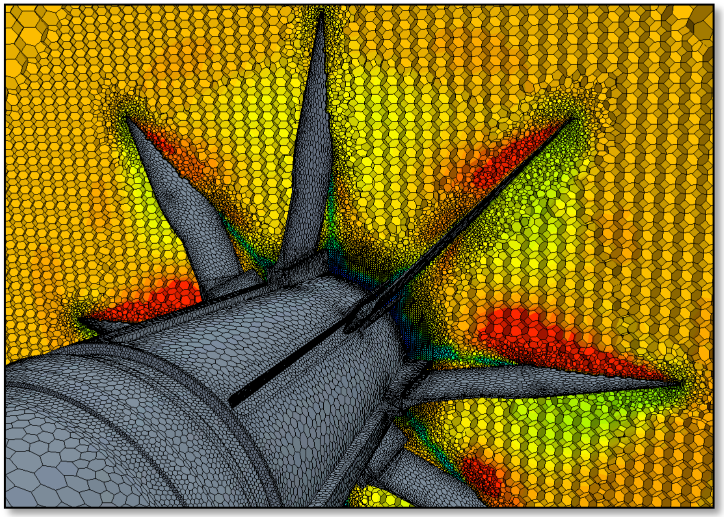
- Advantages
  - Fast, simple grid generation
  - Complex geometries
  - Adaptive grid refinement
  - Fast (~4 hours on 4 cores)
  - In-house (unlimited usage)
- Disadvantages
  - Cartesian grid
  - Limited ability to handle boundary layers
  - External aerodynamics only
  - Marginal overall accuracy in terms of drag and pitching moment



Solvers – Star-CCM+

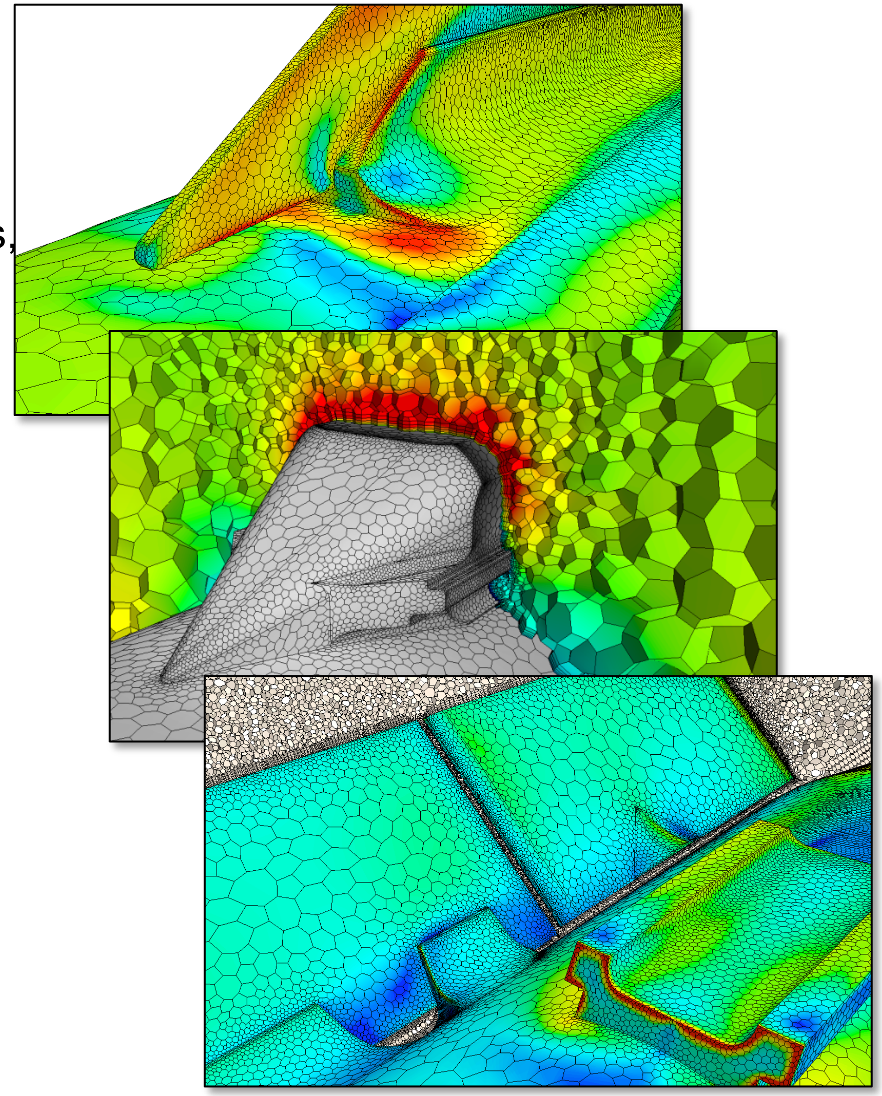
- Advantages
  - Hybrid structured/unstructured body-fitted grids
  - Complex geometries
  - Reasonable grid generation times
  - Good geometry/boundary layer definition.
  - General purpose
  - Improved accuracy (esp. drag, pitching moment)
- Disadvantages
  - No automated adaptive grid refinement
  - Computationally more expensive (~10 hours on 16 cores)
  - Commercial…cost/limited seats

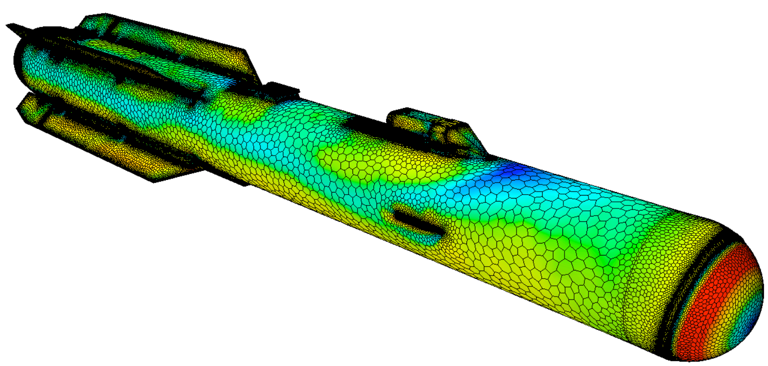




Grid / Computational Domain

- CAD geometry imported in STEP format
  - Surface repair tools used to clean up geometry
  - Many complex protrusions, mounts, holes, steps are retained
- Polyhedral volume mesh
  - Volume sources used to refine mesh in critical areas
  - 5 rows of prism layers near the walls
  - Approximately 4.2 million cells overall
  - Fine mesh with 19.0 million cells used to assess grid independence





Solver Settings

- Density-Based Coupled Solver
  - Steady-state RANS equations
  - SST (Menter) K-w Turbulence Model
    - Wall functions used near the solid boundaries
  - 2nd-order spatial discretization
- Freestream boundary condition applied ~250 diameters from the body
- Uniform flowfield initialization based on freestream conditions

- CPU Time
  - 4 Intel Xeon E5630 (Quad-Core) 3.2GHz CPUs (16 Cores)
  - Approximately 10 hrs per condition

Batch Submission

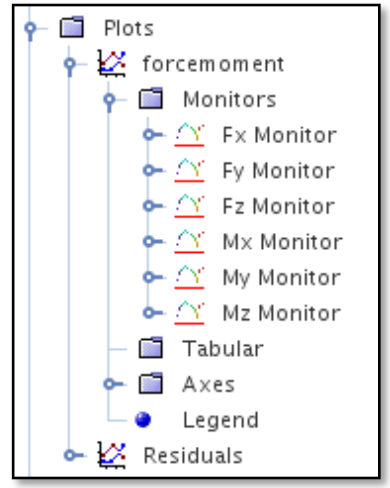
- Jobs are batch-submitted through SGE scheduler
- A Perl script is used as a front-end to generate and submit runs

```perl
#!/usr/bin/perl

#Set user variables
$numproc = 16;
$queue = "f8300";
$submit_dir = "/home/dosnyder/starccm/jcm_test";
$outfile_root = "jcm_test";
$inputsim_name = "jcm_test.sim";

@machs = (0.5, 0.75, 1.25);
@alphas = (0.0, 4.0, 8.0, 12.0, 16.0, 20.0);
@betas = (0.0);

$altitude = 20000; #(feet)
...
#First Order iterations
@cfls1   = (2.0, 10.0, 15.0, 20.0);
@nsteps1 = (20,  20,   20,   60  );

#Second Order iterations
@cfls2   = (2.0, 5.0,  10.0,  15.0,   20.0);
@nsteps2 = (50,  50,   50,    50,     350 );
#End user variables
...
#Loop over the cases
foreach $mach (@machs) {
  foreach $alpha (@alphas) {
    foreach $beta (@betas) {
      #Generate the filename for this case, i.e. "jcm_test_m0.9_a_4.0_b0.0"
      $filename_tag = "_m" . $mach . "_a" . $alpha . "_b" . $beta;
      $filename_current = $outfile_root . $filename_tag;
      ...
      #Generate Star-CCM+ Java macro
      ...
      #Submit job to SGE scheduler
      ...
    }
  }
}
```

- Defines the run matrix
- Defines the free stream temperature & pressure
- Defines the CFL stepping
- Base filename is appended with ‘tokens’ and ‘values’ that define the unique case

Data Reduction

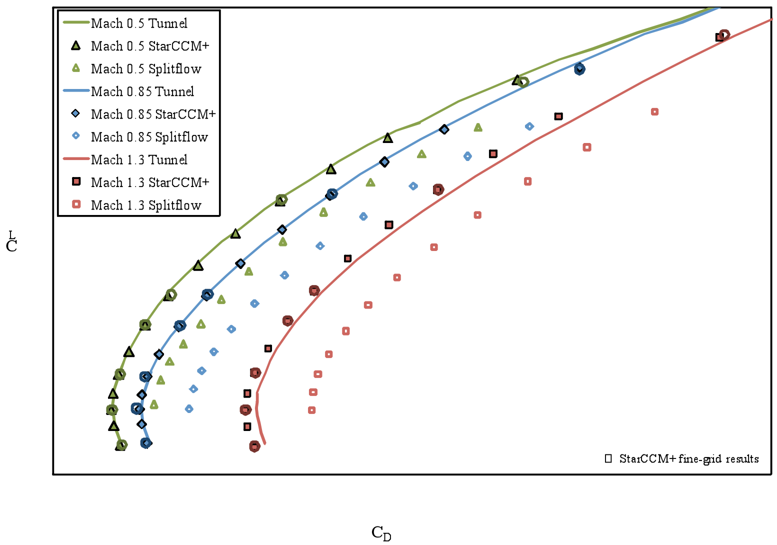
- Force and moment reports / monitors are created and compiled into a single plot object.
  - May include forces / moments for individual components
- Upon completion of the run, the Java macro exports the plot values to a data file.
  - Unique file name, including ‘tokens’ and ‘values’
  - May include wing sweep angles, control surface deflections, etc.
- To reduce the data, a script is executed that
  - Loops through the output files
  - Determines the flight conditions
  - Averages the last *n* iterations in the file
  - Generates a single tabular data file

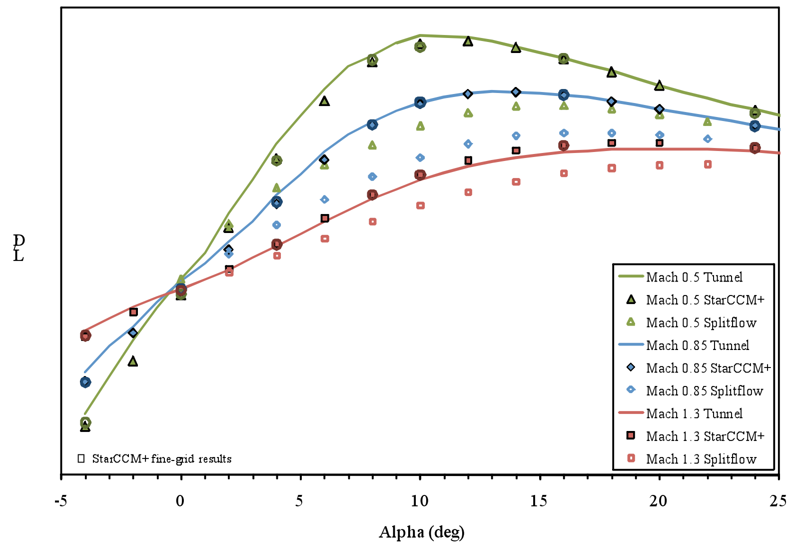


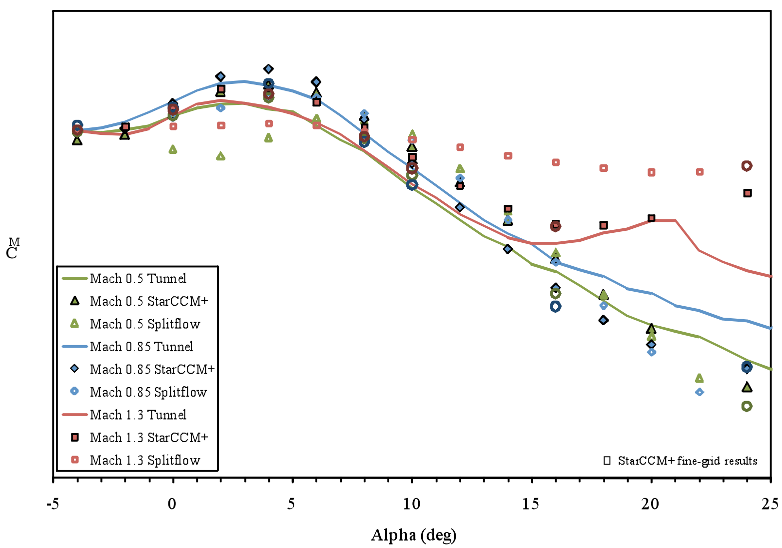

```
jcm_test_m0.5_a0.0_b0.0.dat
jcm_test_m0.5_a4.0_b0.0.dat
jcm_test_m0.5_a8.0_b0.0.dat
jcm_test_m0.5_a12.0_b0.0.dat
jcm_test_m0.5_a16.0_b0.0.dat
jcm_test_m0.5_a20.0_b0.0.dat
jcm_test_m0.75_a0.0_b0.0.dat
...
jcm_test_m1.25_a16.0_b0.0.dat
jcm_test_m1.25_a20.0_b0.0.dat
```

Mach	alpha (deg)	beta (deg)	Fx (lbf)	Fy (lbf)	Fz (lbf)	Mx (lbf-ft)	...
0.500000E+00	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00	...
0.500000E+00	0.400000E+01	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00	...
0.500000E+00	0.800000E+01	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00	...
...	...	...	...	...	...	...	...

Aerodynamic Forces/Moments







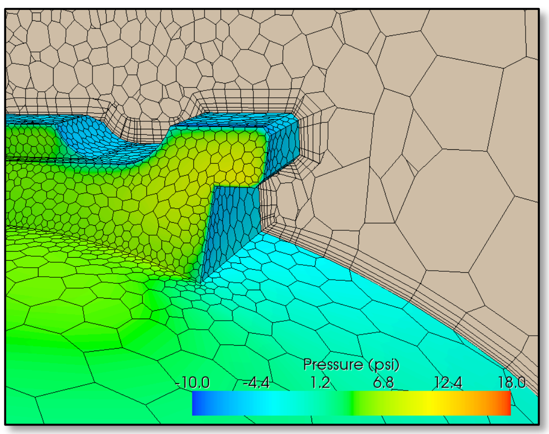
- Aerodynamic forces and moments are predicted well using Star-CCM+
  - Lift / Drag within ~3%
  - Trim angle within ~1°
- Star-CCM+ results are significantly improved over Splitflow solver

Mesh and Turbulence Model Study

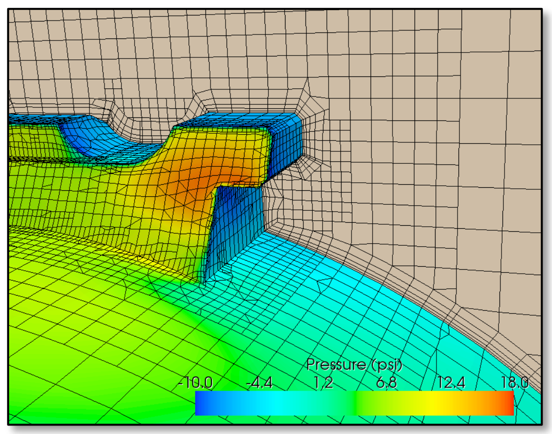
	Cell Type	Cells	Faces	Prism Layers	Wall y^+	Turb. Model
Baseline	Poly	4.2M	23.9M	5	~75	SST K-w
Trimmer	Trim	8.8M	26.5M	5	~75	SST K-w
Low y+	Poly	8.6M	40.4M	25	~1	S-A

* All three meshes utilize the same surface sizing parameters

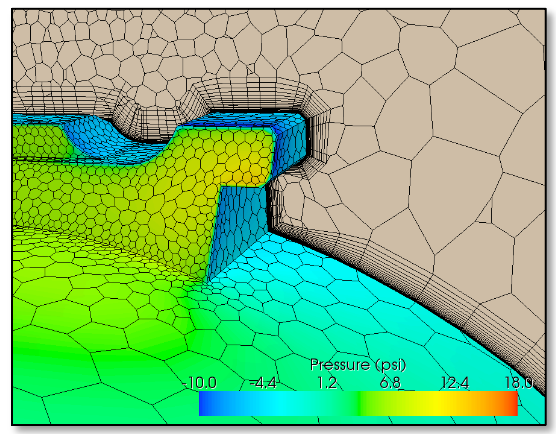
* Baseline and Trimmer mesh have nominally the same number of cell faces



Baseline

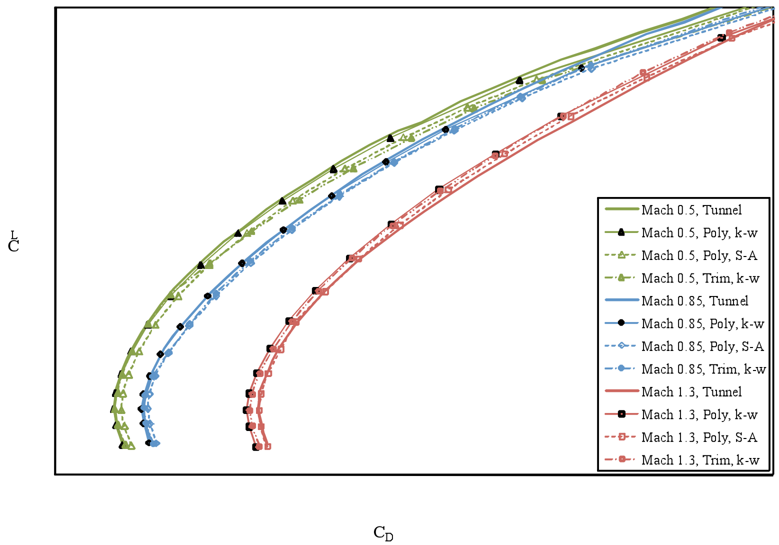


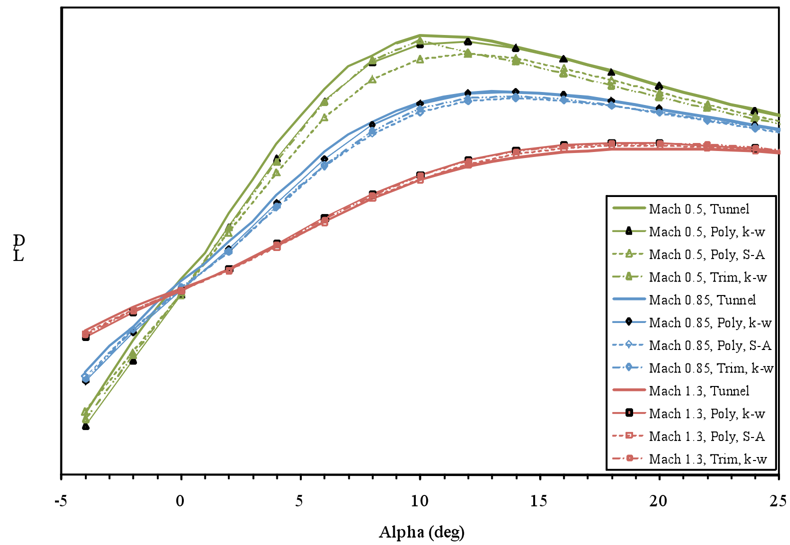
Trimmer

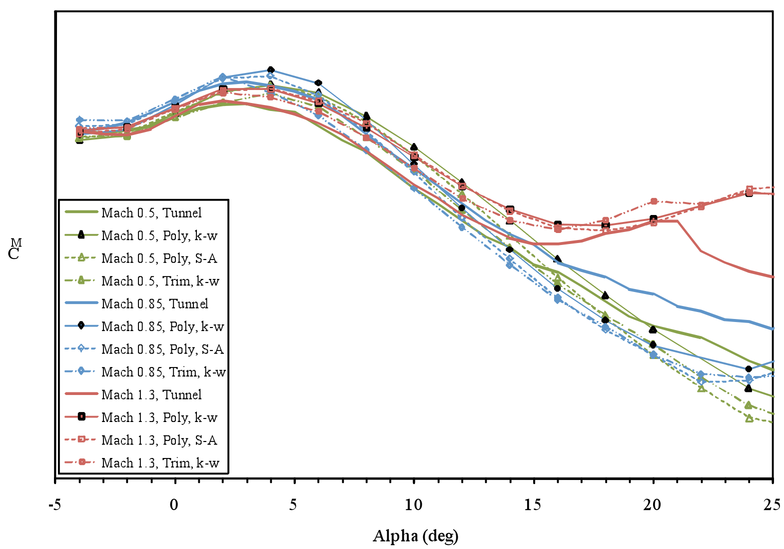


Low y^+

Aerodynamic Forces/Moments



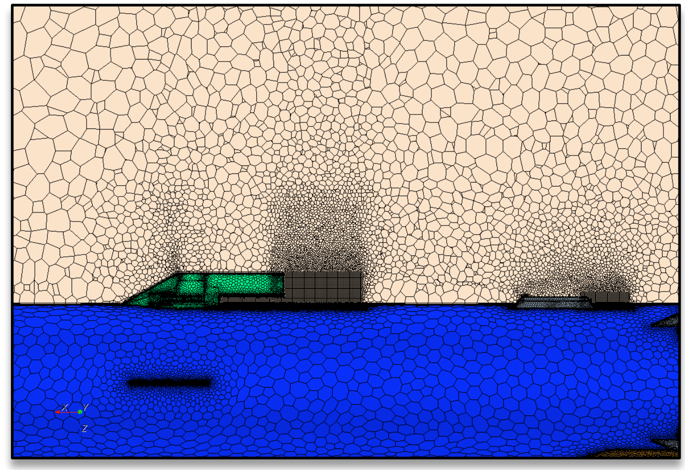


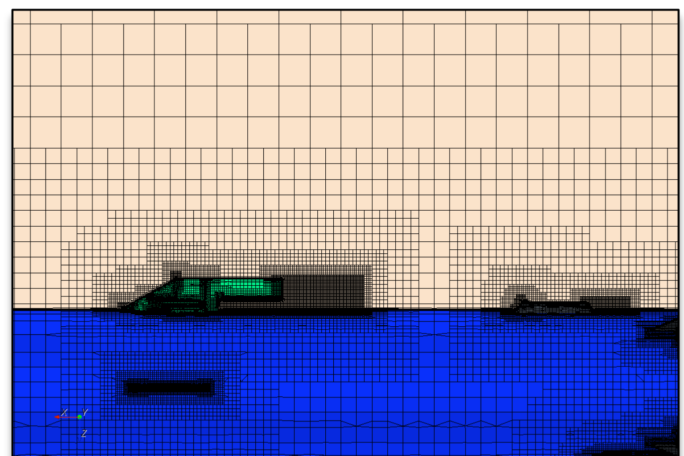


- Turbulence model
  - SST K-w model w/wall functions provides best results for subsonic conditions.
  - S-A model integrated to the wall provides best results for supersonic conditions.
- Mesh type
  - Trimmer / Polyhedral meshes produce similar results at low angles of attack.
  - Polyhedral mesh produces better results at higher angles of attack

Mesh Discussion

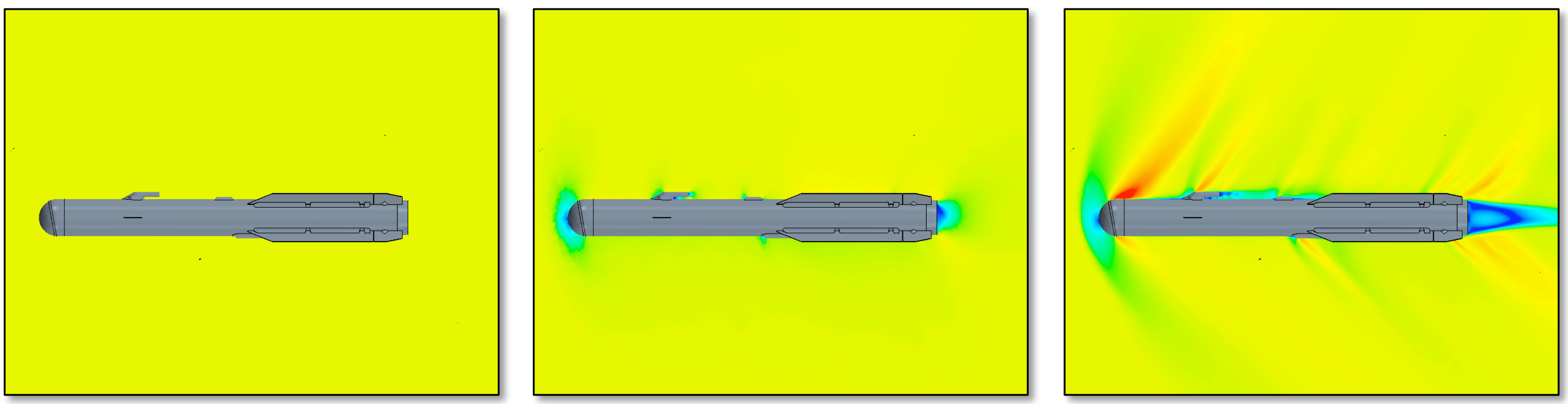
- Mesh behavior may be due to:
  - Polyhedral mesh has more random orientation of faces, yielding similar numerical dissipation at all angles of attack.
  - Polyhedral mesh tends to place many cells radially away from the body, which may help at higher angles of attack.





Solution Acceleration – Initialization

- Uniform Initialization
  - Domain is uniformly initialized to the freestream conditions
  - A linear reduction to zero-velocity is applied near the walls based on a user-specified wall distance.
- Grid Sequencing Initialization
  - Available in Star-CCM+ V5.04
  - Provides a better initial condition by solving for an approximate inviscid solution via a series of coarsened meshes.
    - Takes ~1-2 minutes for the baseline JCM mesh
  - Allows more aggressive CFLs early in the solution



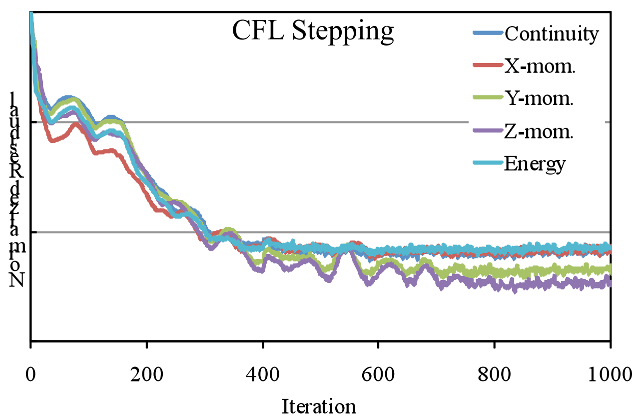
Uniform Initialization | Grid Sequencing Initialization | Final RANS Solution

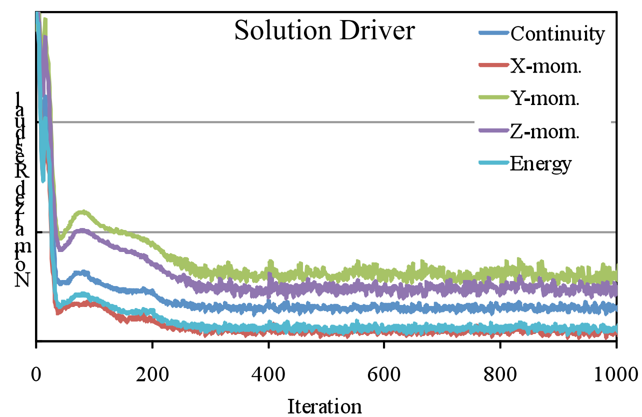
Solution Acceleration – CFL Control

- CFL Stepping (Our Legacy Approach)
  - User-defined via Java

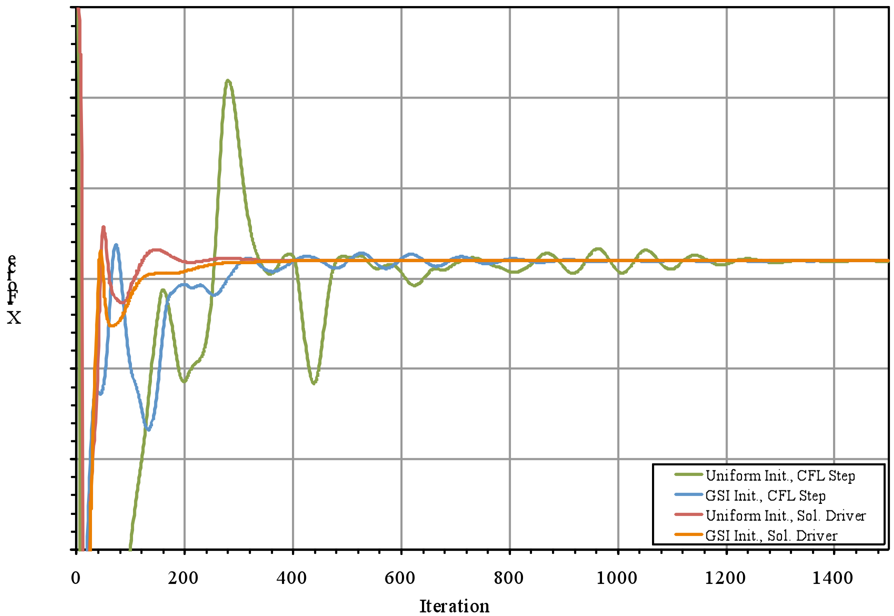
CFL	2.0	3.0	6.0	9.0	12.0
Iterations	150	250	250	200	650

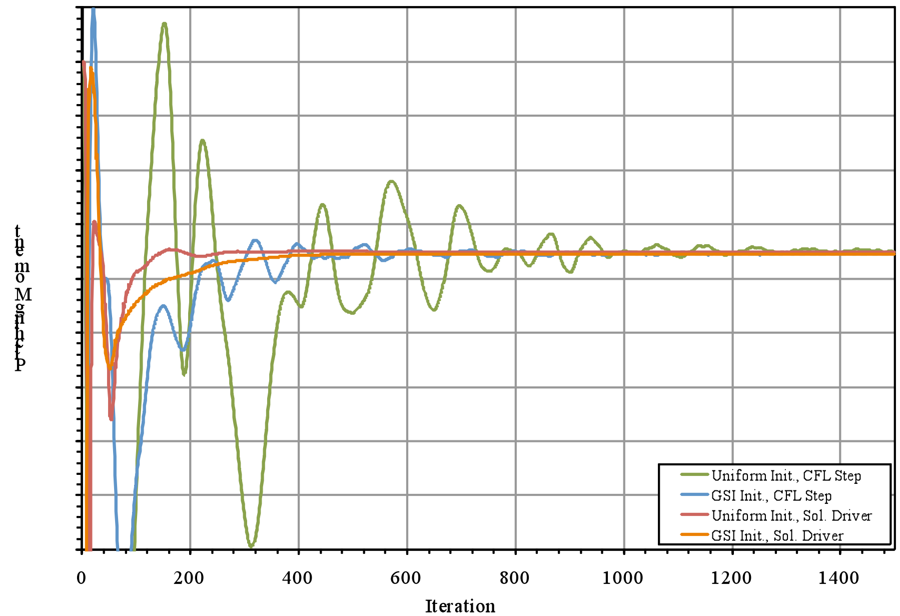
  - Lower Mach numbers allow higher CFLs
    - Divide the number in the CFL stepping by the Mach number
    - Works well for Mach 0.5-2.5
- Solution Driver
  - Available in V5.06
  - Combines a CFL ramp with corrections control/limiting
  - Provides a straight-forward and robust convergence acceleration





Solution Acceleration Results
Mach 0.85





- GSI significantly improves convergence rate for CFL Stepping.
- Solution Driver provides best results
  - Oscillations about converged value are reduced
  - Uniform Initialization provides slightly faster convergence

Conclusion

- Accuracy of results
  - Star-CCM+ solutions provide a significant improvement over our in-house code at predicting external aerodynamic forces and moments.
  - Both Star-CCM+ and Splitflow are currently integrated into our analysis procedures
    - Splitflow: Preliminary analyses/trades, large run matrices
    - Star-CCM+: Refined analyses, drag-critical, internal/external flows, conjugate heat transfer, LES, etc.

- Mesh/Solver options
  - For our typical application at transonic/supersonic Mach numbers
    - Polyhedral meshes with ~5 prism layers and 4M cells
    - SST k-w turbulence model with wall functions
    - Grid Sequencing Initialization combined with Solution Driver CFL control provides a robust method to achieve converged solutions at a computational savings of 20-50% over manual CFL ramping.

- Automation of solving/post-processing using Perl and Java reduces user interaction to only pre-processing stages, reduces user-error, and increases throughput.