Recent Enhancements to OVERFLOW for Spatial and Temporal Adaption

Pieter G. Buning

NASA Langley Research Center, Hampton, VA

and

Thomas H. Pulliam

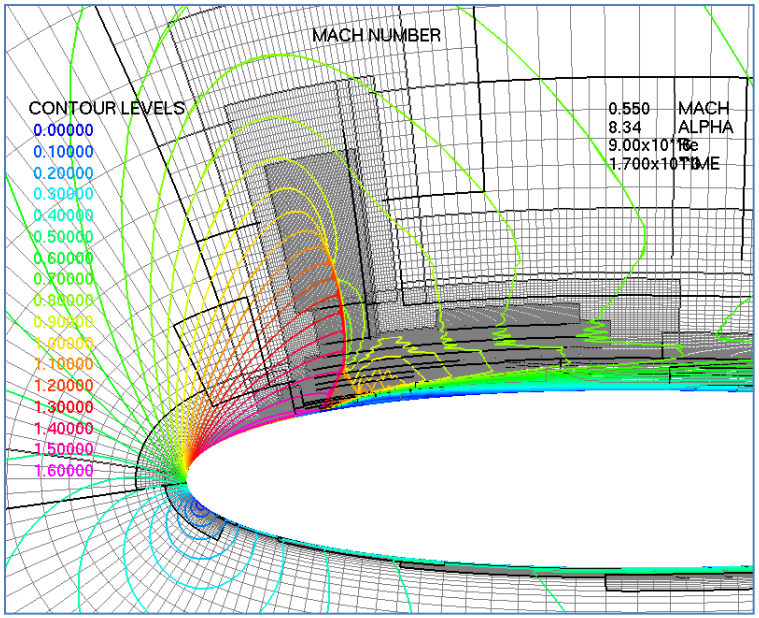
NASA Ames Research Center, Moffett Field, CA

12th Symposium on Overset Composite Grid and Solution Technology

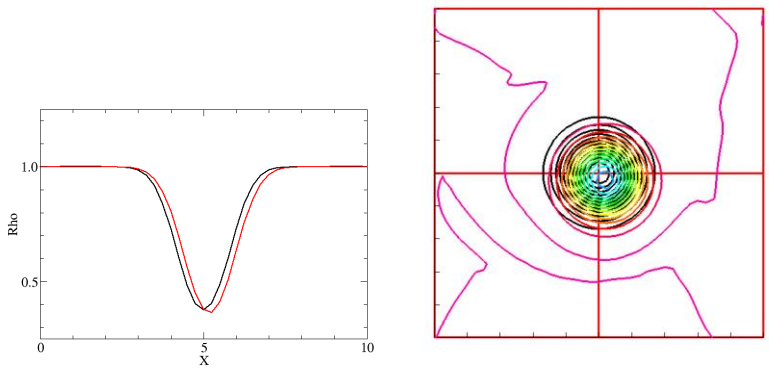
October 6-9, 2014, Atlanta, GA

Topics

- Improvements to grid adaption process
  - Added controls
  - Improved robustness



- New time-advance schemes/controller
  - Multi-step and multi-stage schemes
  - Temporal error controller

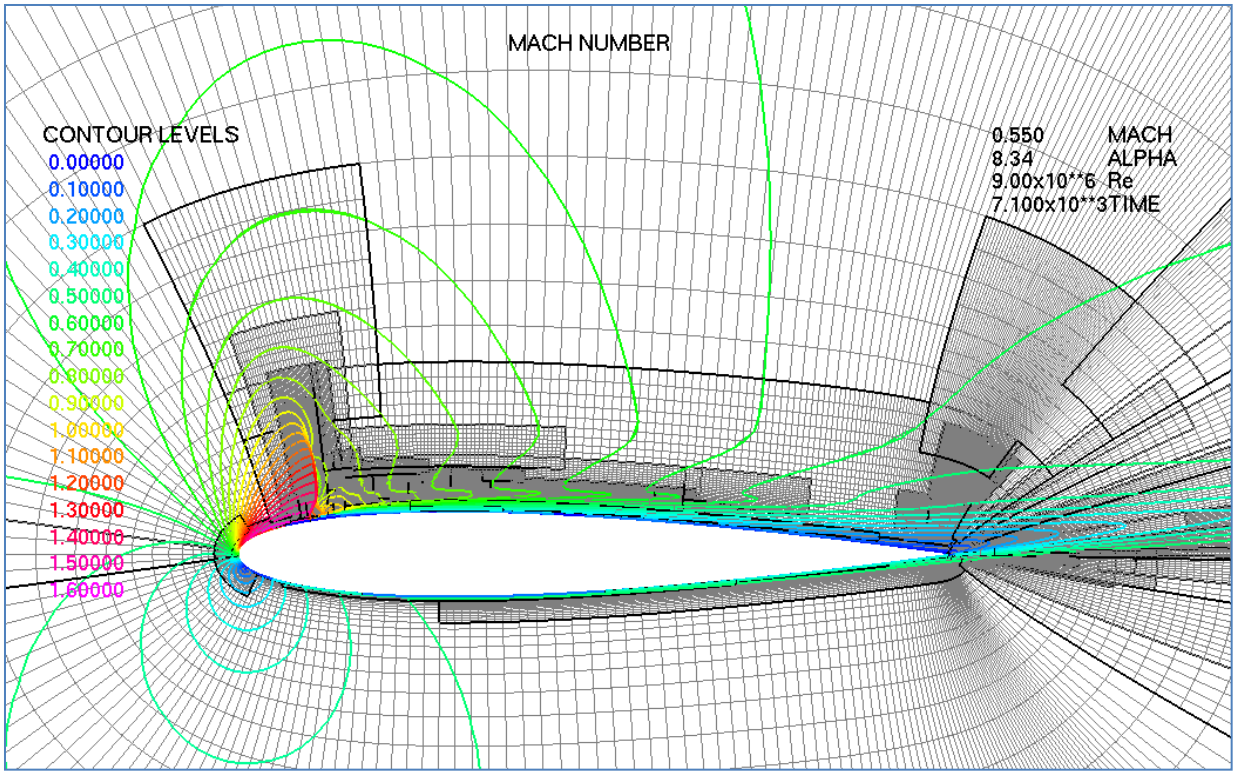


Δt = 0.1

Grid Adaption Process - Review

- Error estimation
  - Compute sensor function at each grid point
    - 2nd undivided difference of Q variables: $\max_{i=j,k,l}\left\{\left[\frac{\frac{1}{2}(q_{i-1}-2q_i+q_{i+1})}{q_{ref}}\right]^2\right\}$
  - Mark each 8x8x8 box for refinement, coarsening, or maintaining grid level (regions can only change by one level per adapt)
  - Merge neighboring boxes and send to manager
- Grid generation
  - Manager collects boxes, adds buffers so neighboring regions only differ by one refinement level
  - Finer regions blank out underlying coarser grid levels
  - Boundary communication uses specialized Chimera interpolation stencils

Grid Adaption Process - Review



- Grid adaption issues:
  - Too many/unpredictable number of grid points
  - Responds to spurious or small disturbances

Controlling Grid Growth*

- Want two controls:
  - Maximum grid size (represents cost, or time per step)
  - Maximum growth rate (makes adaption process more robust)

  We can implement both using the same process

*Following concepts of M. Nemec and M.J. Aftosmis, "Toward Automatic Verification of Goal-Oriented Simulations," July 2014, chapter contribution to JANNAF Guide to Uncertainty Quantification and Simulation Credibility for Continuum Physics Applications, and M. Nemec, M.J. Aftosmis, and M. Wintzer, "Adjoint-Based Adaptive Mesh Refinement for Complex Geometries," AIAA 2008-0725, Jan. 2008.

Controlling Grid Growth

- Process:
  - For each grid:
    - Compute sensor function
    - Store 8x8x8 point boxes with maximum sensor function value
  - Calculate current and maximum new grid size (limited by max size and growth rate)
  - Newton's method: relax the refine and coarsen error tolerances to give the desired grid size*
    - Mark boxes as refine, maintain, or coarsen (1,0,-1)
    - Merge neighboring boxes and send to manager
    - Manager generates box description of near- and off-body grids
    - Calculate resulting grid size, update tolerance offset, iterate

*This process is similar to S. Peron and C. Benoit, "Automatic Off-Body Overset Adaptive Cartesian Mesh Method Based on an Octree Approach," AIAA 2011-3050, June 2011.

Controlling Grid Growth

- Some nice properties of this process:
  - Newton's method converges!
  - Little work inside the Newton loop
  - Very little work done by manager only
  - Minimal data communication (boxes converted to (-1,0,1), merged)
  - Does not involve grid splitting or hole cutting processes

Sample diagnostic output:

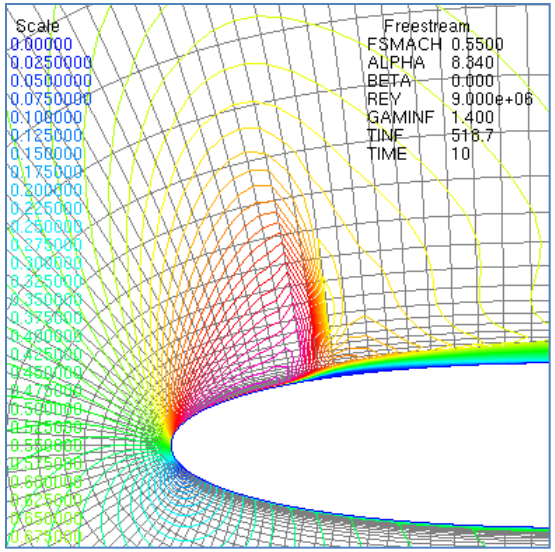
```
Maximum grid size is limited by growth factor.
Current grid size(K):          312.
Maximum grid size(K):          405.

Threshold shift      Grid size(K)   Growth factor
   0.00                 488.           1.56
  -1.00                 305.           0.98
  -0.45                 405.           1.30
```

Additional Improvements

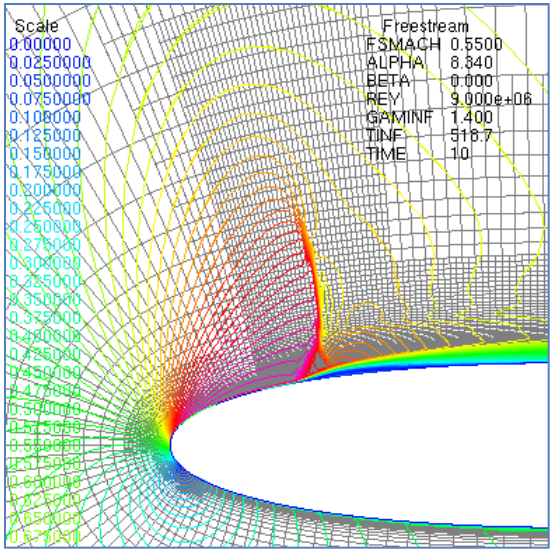
- Sensor function smoothing
  - Eliminate local peaks and ridges
- Distance weighting of sensor function
  - Can specify a distance from the body which represents a one-refinement-level decay of the sensor function
  - Additional parameter sets the distance to start the decay

Example: NACA 0012

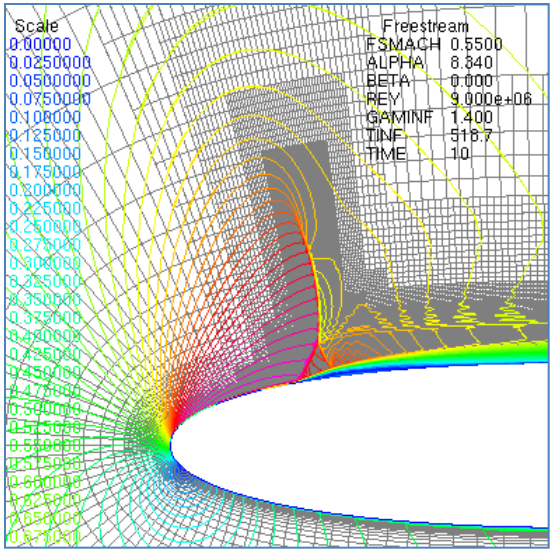
- Mach 0.55, α=8.34˚, Re=9M/c
- Original grid 19K pts, final grid 330K pts
- 20 adapt cycles with 2 levels of refinement
- 20 adapt cycles with 4 levels of refinement



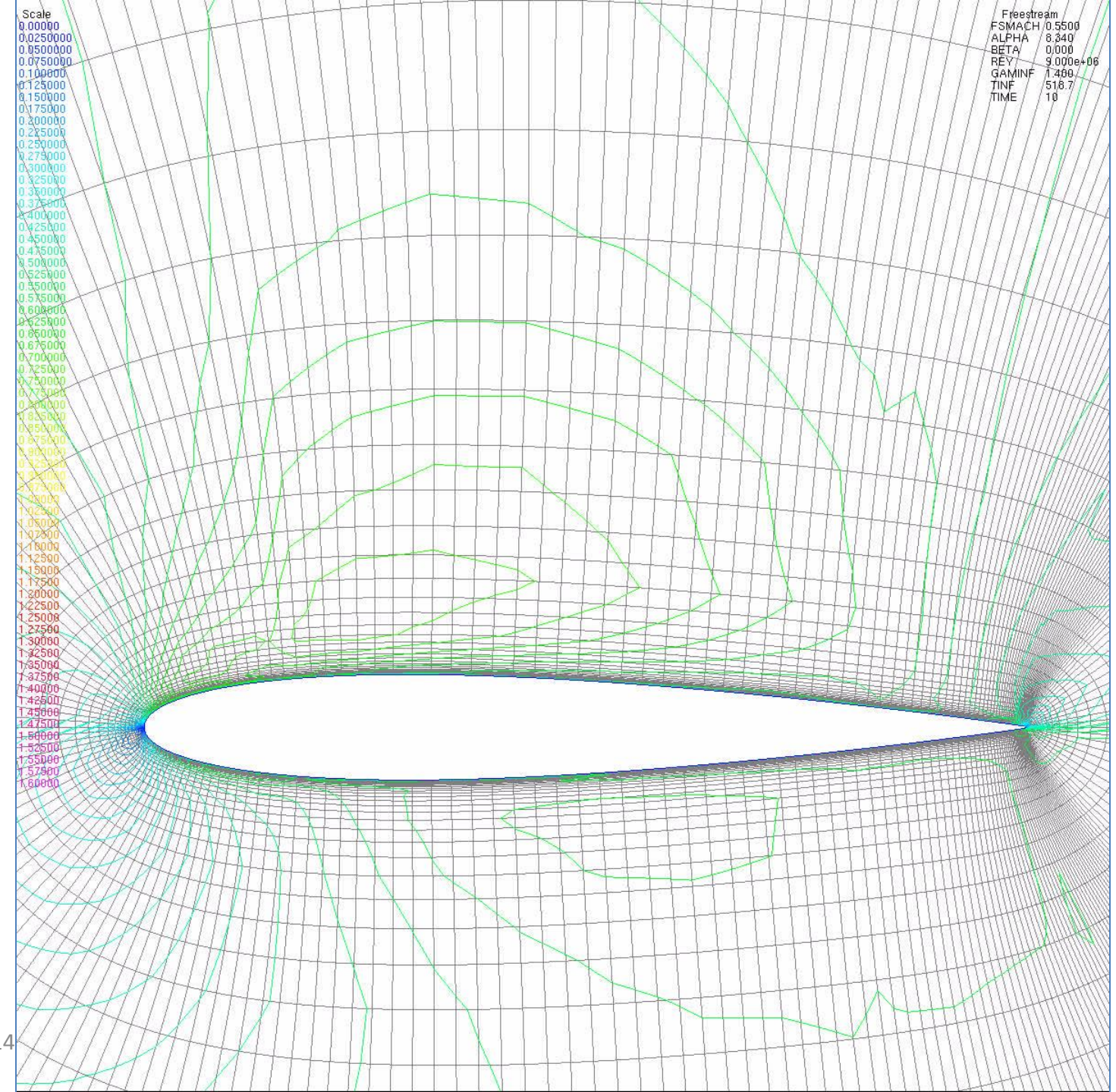
300 Steps
CL=0.925



500 Steps
CL=0.954

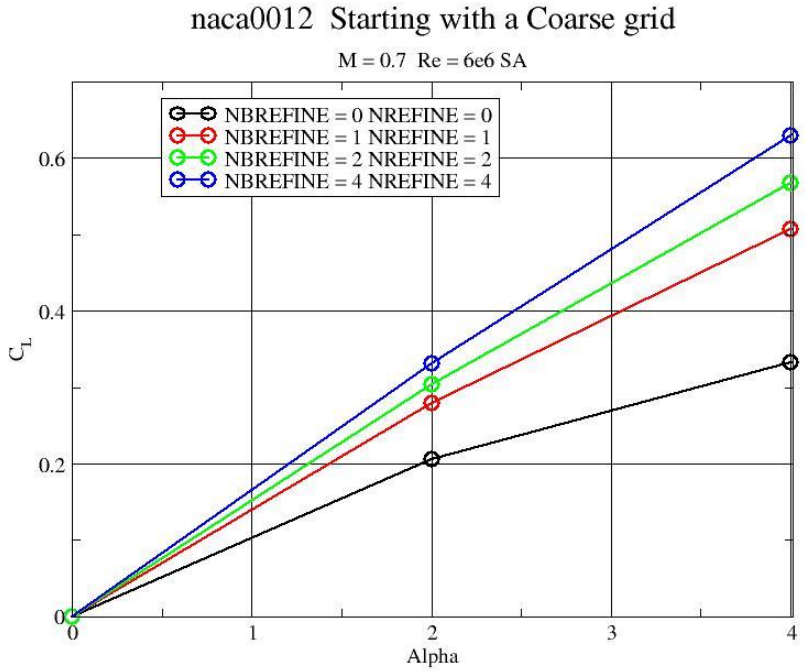


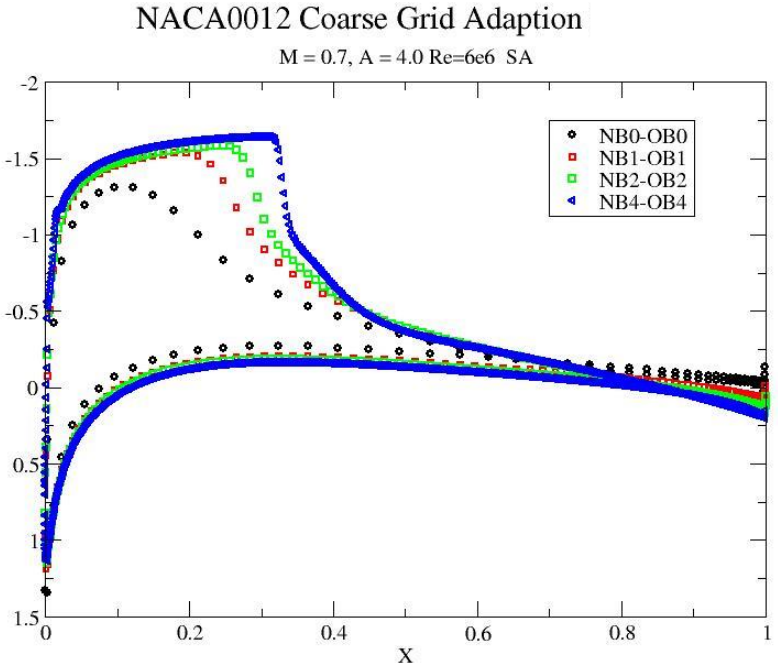
900 Steps
CL=0.950


Scale
0.00000
0.0250000
0.0500000
0.0750000
0.100000
0.125000
0.150000
0.175000
0.200000
0.225000
0.250000
0.275000
0.300000
0.325000
0.350000
0.375000
0.400000
0.425000
0.450000
0.475000
0.500000
0.525000
0.550000
0.575000
0.600000
0.625000
0.650000
0.675000
0.700000
0.725000
0.750000
0.775000
0.800000
0.825000
0.850000
0.875000
0.900000
0.925000
0.950000
0.975000
1.00000
1.02500
1.05000
1.07500
1.10000
1.12500
1.15000
1.17500
1.20000
1.22500
1.25000
1.27500
1.30000
1.32500
1.35000
1.37500
1.40000
1.42500
1.45000
1.47500
1.50000
1.52500
1.55000
1.57500
1.60000
Freestream
FSMACH 0.5500
ALPHA 8.340
BETA 0.000
REY 9.000e+06
GAMINF 1.400
TINF 518.7
TIME 10

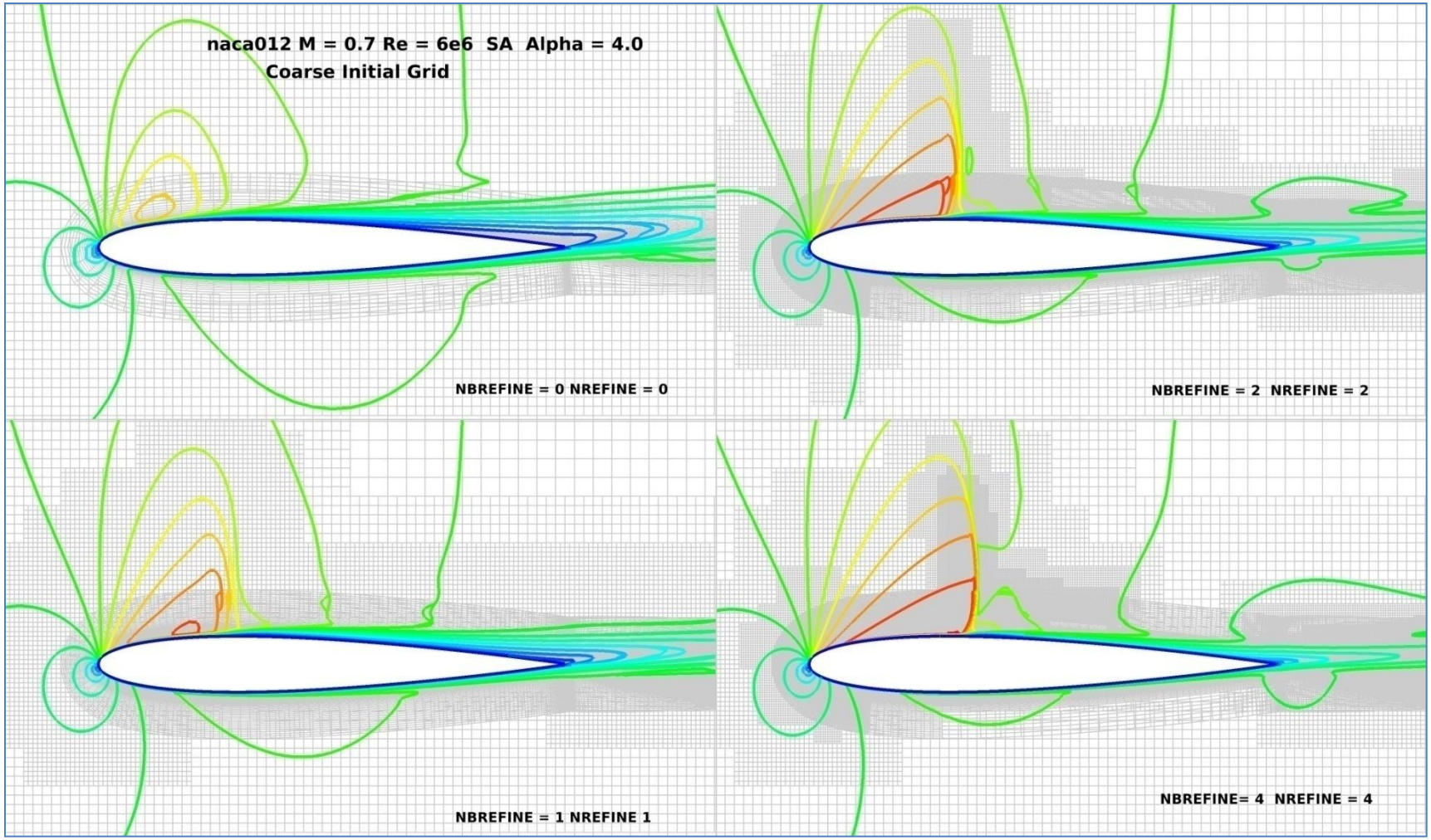
Example: NACA 0012 Polar

- Start with a very coarse grid
  - Near-body grid has 51 pts per side, 33 pts in viscous direction
  - Off-body grids have minimum 2%c spacing
  - Flow conditions: Mach 0.7, Re_c=6M



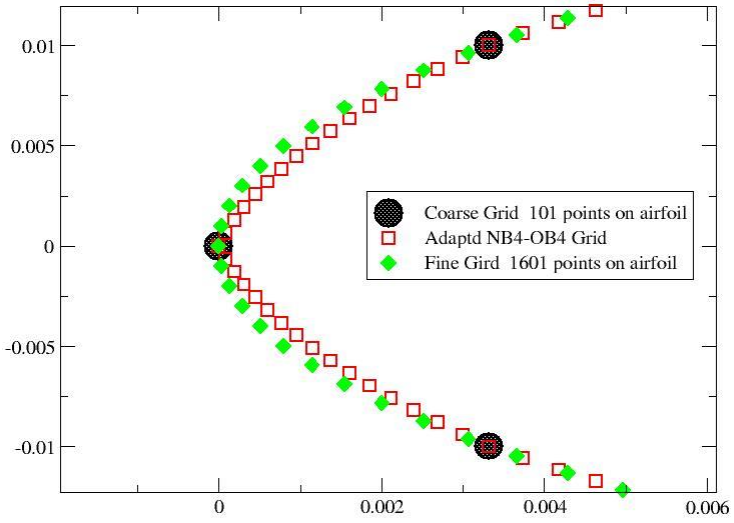


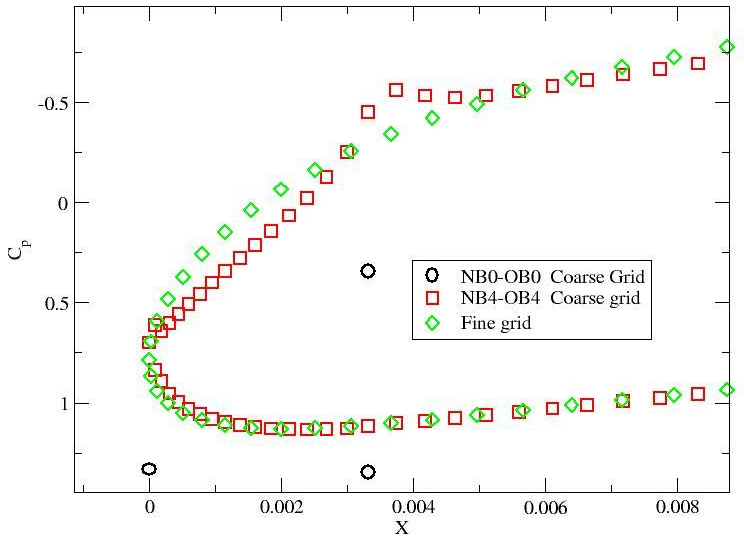
Example: NACA 0012 Polar



Example: NACA 0012 Polar

- Problem at leading edge: grid is so coarse, adaption scheme thinks it might be a sharp corner



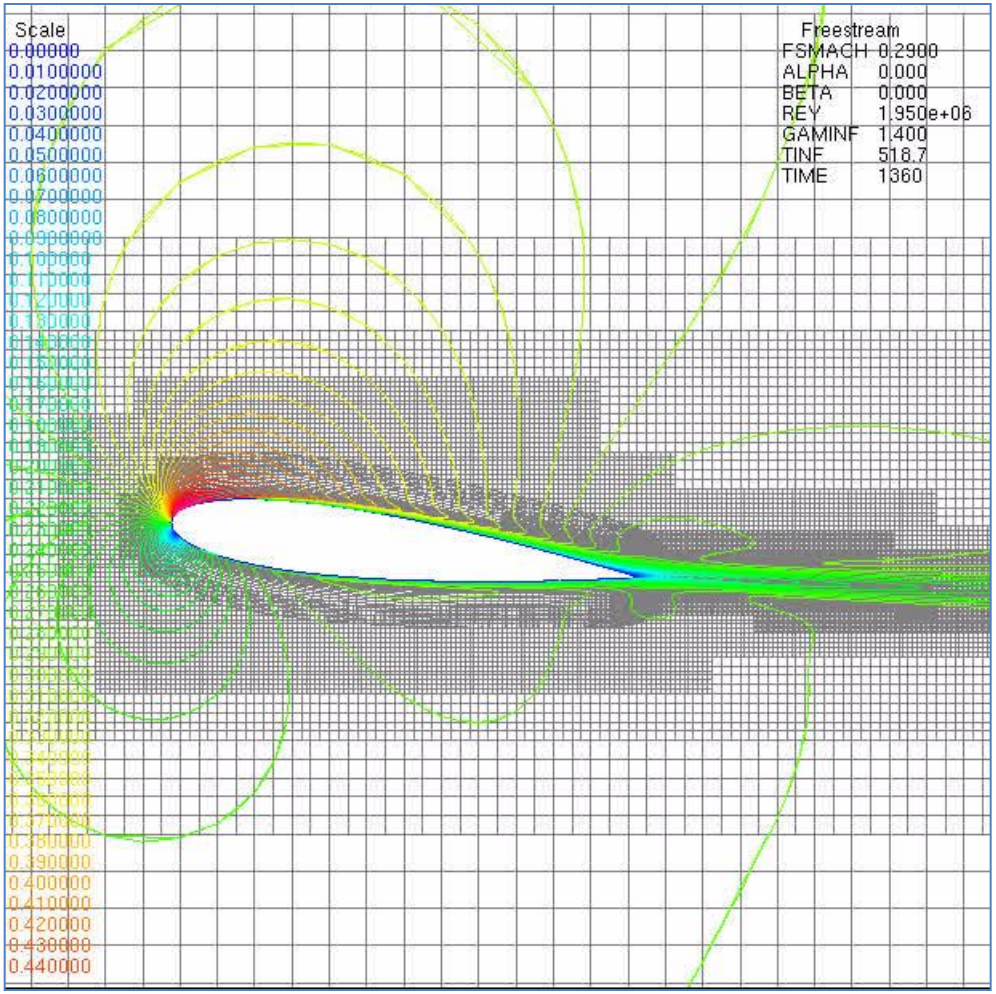


Example: Pitching Airfoil

NACA 0015, Mach 0.29, Re=2M/c,
α=11˚±4.2˚

(Piziali Case 2, mild stall)

- 4 levels of adaption (near-body and off-body)
- Adaption every 10 steps
- 360 steps per pitch cycle, 30 subiterations/step



Time-Advance Methods – The Current Lineup

- Multi-step methods: BDF1, BDF2, BDF2OPT
  - BDF2OPT is a blend of BDF2 and BDF3
  - Officially 2nd-order in time, but "half the error" of BDF2
  - Uses 3 time levels (q^n, q^{n-1}, q^{n-2}), but almost no extra work

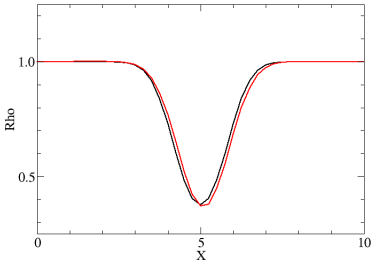
  Ref: V.N. Vatsa, M.H. Carpenter, and D.P. Lockard, "Re-evaluation of an Optimized Second Order Backward Difference (BDF2OPT) Scheme for Unsteady Flow Applications," AIAA 2010-0122, Jan. 2010
- Multi-stage methods: RK3, RK4, ESDIRK3, ESDIRK4
  - ESDIRK4 is 4th-order in time, but has 5 stages
  - Each stage requires subiterations

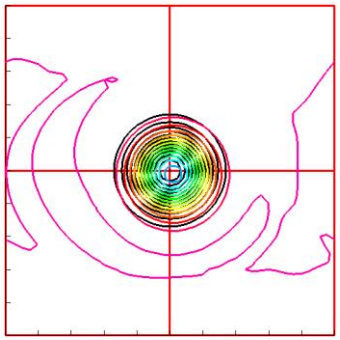
  Refs: M.H. Carpenter, C.A. Kennedy, H. Bijl, S.A. Viken, and V.N. Vatsa, "Fourth-Order Runger-Kutta Schemes for Fluid Mechanics Applications," J. Scientific Computing, Vol. 25, Nos. 1/2, Nov. 2005; and T.H. Pulliam, "Time Accuracy and the Use of Implicit Methods," NASA Ames Research Center, 2014

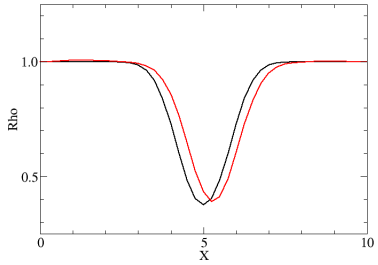
2D Inviscid Vortex Convection – BDF2OPT

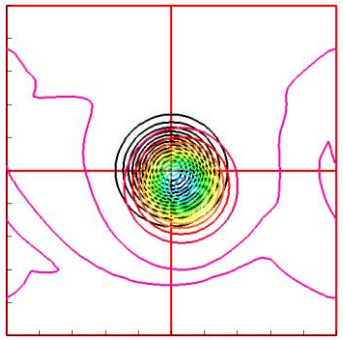
- BDF2OPT gives a better answer than BDF2, but is still 2nd-order accurate (has essentially the same limit on time-step)
- Cost is the same

BDF2



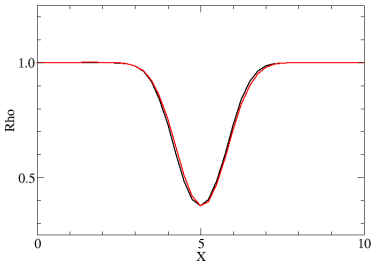


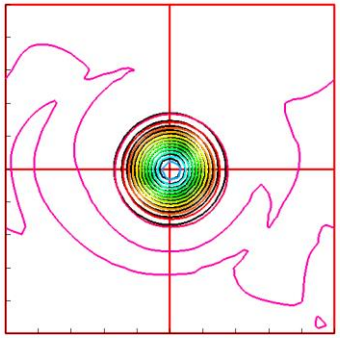


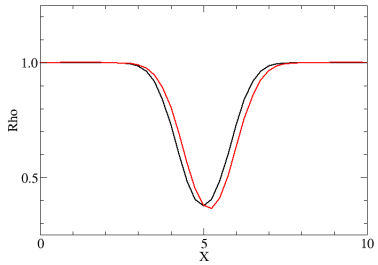


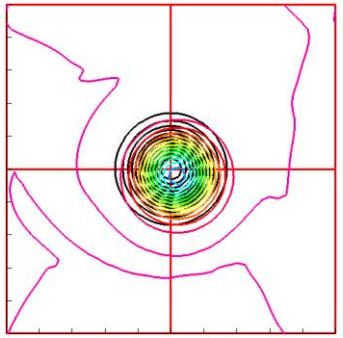
Δt = 0.05 Δt = 0.1

BDF2OPT







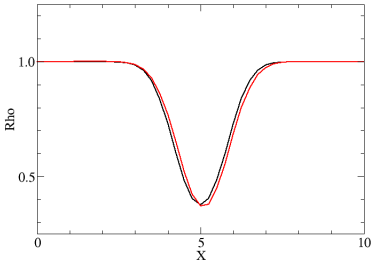


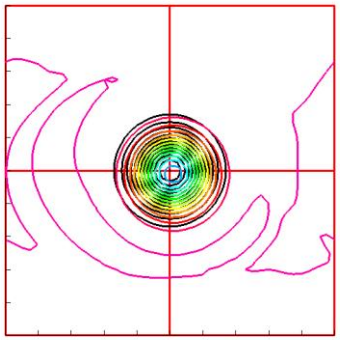
Δt = 0.05 Δt = 0.1

2D Inviscid Vortex Convection – ESDIRK4

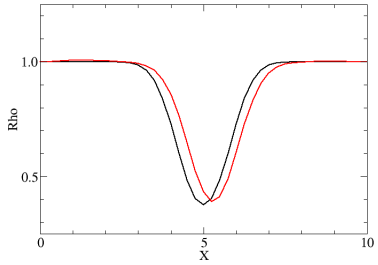
- ESDIRK4 can take 10x the time-step of BDF2 (but takes 5x the work)!
- This case is ideal—on real problems ESDIRK4 is much less robust than BDF2

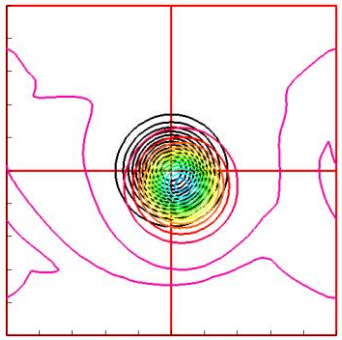
BDF2





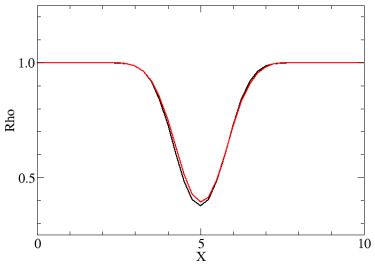
Δt = 0.05

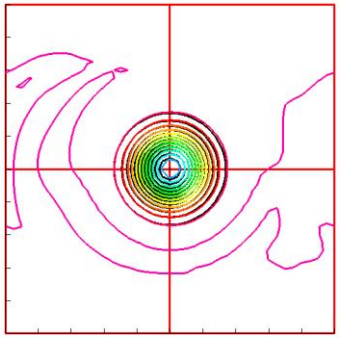




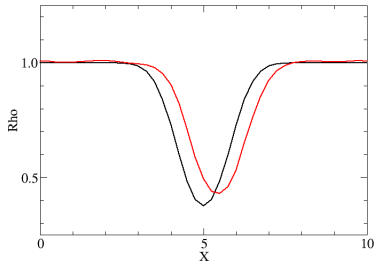
Δt = 0.1

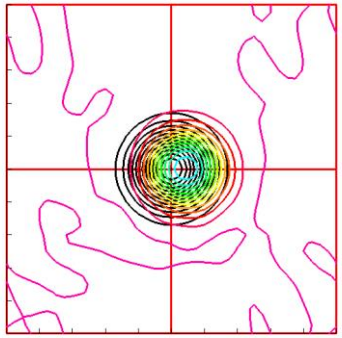
ESDIRK4





Δt = 0.5





Δt = 1.0

Temporal Error Controller*

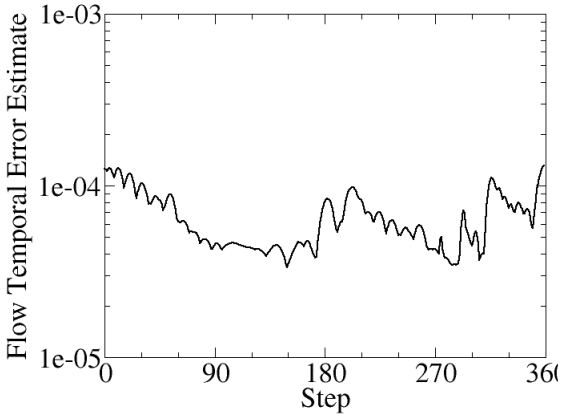
- Allows possible early termination of Newton/dual time-stepping subiterations
  - This would be a big benefit for ESDIRK4, significant benefit for BDF schemes too
- For ESDIRK schemes, error estimate is the L_∞-norm of the difference between design scheme and embedded lower-order scheme residuals at the end of the previous step
- For BDF2OPT scheme, error estimate is the L_∞-norm of the difference between BDF3 and BDF2 residuals at each subiteration
- We aim to converge the subiteration process until the L_∞-norm of the residual is a specified number of orders below the error estimate

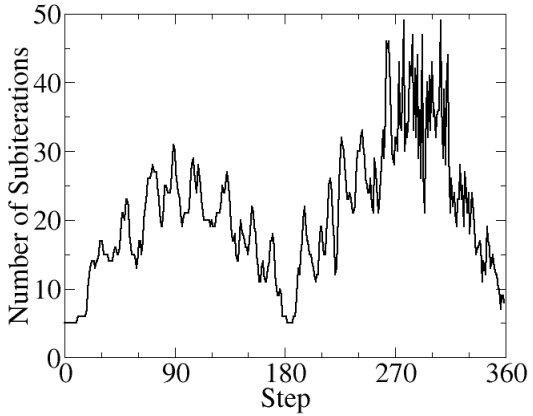
*Ref: M.H. Carpenter, C.A. Kennedy, H. Bijl, S.A. Viken, and V.N. Vatsa, "Fourth-Order Runge-Kutta Schemes for Fluid Mechanics Applications," J. Scientific Computing, Vol. 25, Nos. ½, Nov. 2005.

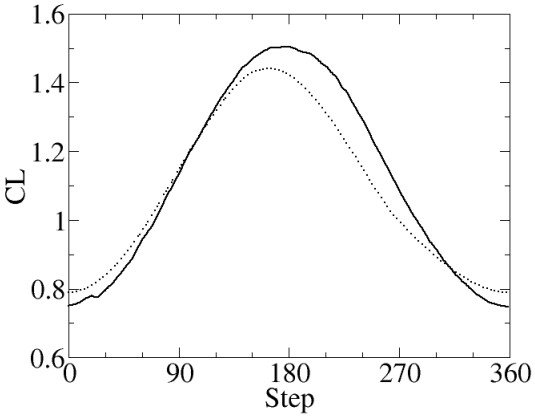
Example: 2D Pitching Airfoil

NACA 0015, Mach 0.29, Re=2M/c, α=11˚±4.2˚ (Piziali Case 2, mild stall)

- 360 steps per pitch cycle, nominally 100 subiterations per step
- Temporal error estimate does not vary much
- Number of subiterations to meet tolerance varies from 5 to 50
- But CL does not match case with full 100 subiterations (dotted line)







Temporal Error Controller

- Questions
  - Should error estimate include volume scaling that is in residual?
  - Should it include turbulence model error estimate?

- Some notes
  - Error estimate from a 4^{th}-order scheme (ESDIRK4) is much smaller that that of a 2^{nd}-order scheme (BDF2OPT), for the same Δt. This should not be a surprise—but it is much harder to get the subiteration residual below this value. A case with comparable error would have a larger Δt.
  - The temporal error estimate can also be used to suggest changes to Δt. For this we need to specify an error tolerance to compare to the error estimate. But our error estimate uses residuals, which have a volume scaling (large residuals in large cells, etc.), so we have no intuition for picking a physical error tolerance.

Summary

- Improvements to grid adaption process give smooth and robust adaption with controlled growth
- New time-advance options are available, and a temporal error controller has been implemented

Still to work on...

- Grid adaption “converges,” except for shocks. Need to be able to mark shocks and limit adaption.
- Need to learn how to use temporal error controller reliably (automatically).
- Can we make good use of ESDIRK4, or do we need a different 4^{th}-order scheme?