



A Parallel Symmetric Successive Overrelaxation Method for OVERFLOW

Joseph M. Derlaga

Charles W. Jackson

Pieter G. Buning

NASA Langley Research Center

CFD-15, Overset Methods

August 4th, 2021



Motivation

- Explicit overset/block boundaries are an obvious impediment to convergence.
- What can be done to build on the convergence improvements introduced in OVERFLOW 2.3 that will decrease time to solution?



Outline

- Background
- Description of the Parallel SSOR Algorithm
- Selected Examples
- Conclusions



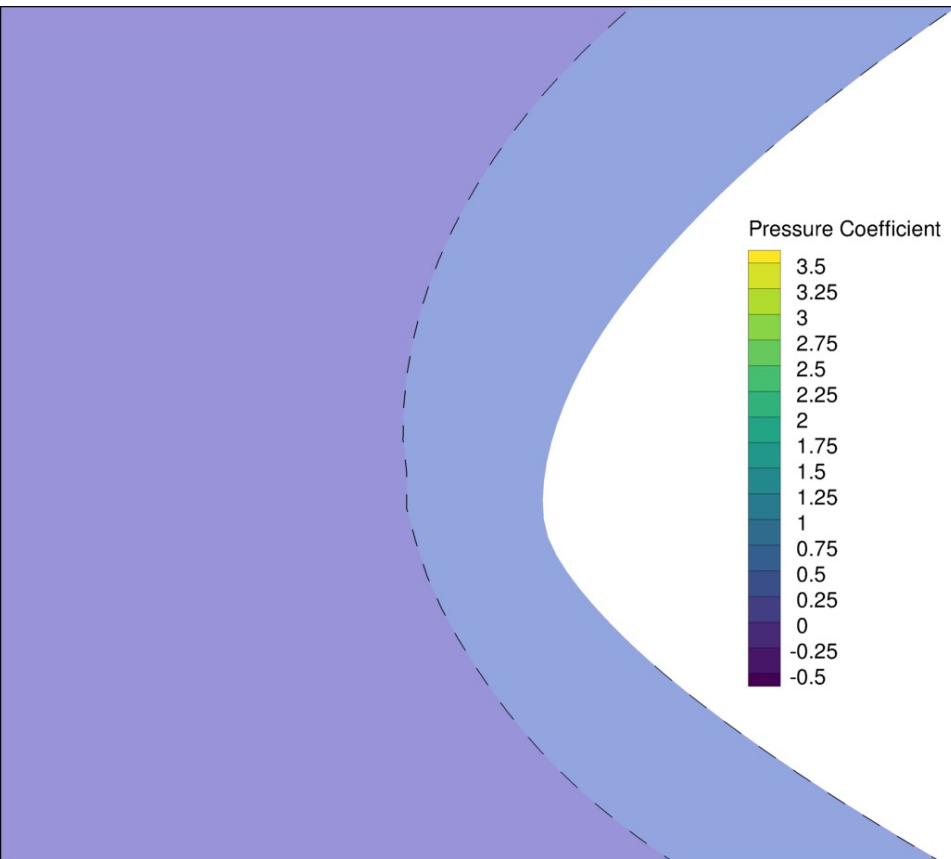
Background

- OVERFLOW 2.3 introduced improved linearizations and the implicit treatment of some physical boundary conditions for the SSOR schemes
 - Please see AIAA paper 2020-1045 for details
- Convergence was greatly improved, but overset and split block boundaries retained their explicit treatment
 - Latent problems are more likely to be encountered as a result of the explicit boundaries

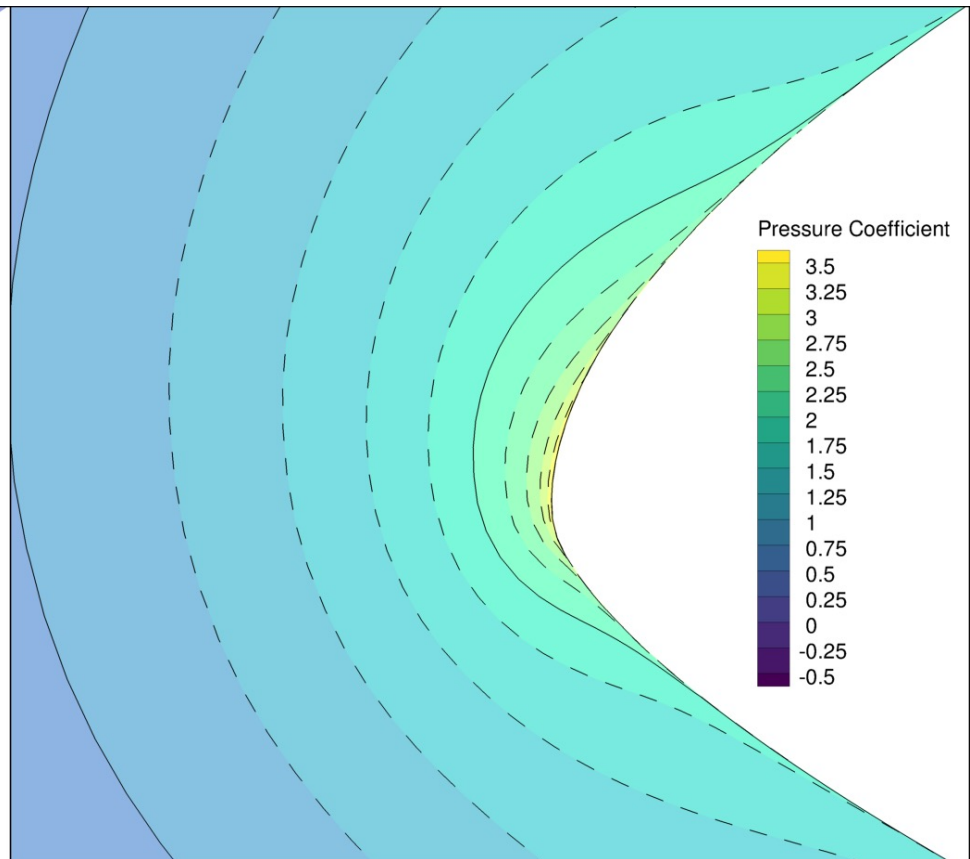


Background

Original SSOR, 1 processor



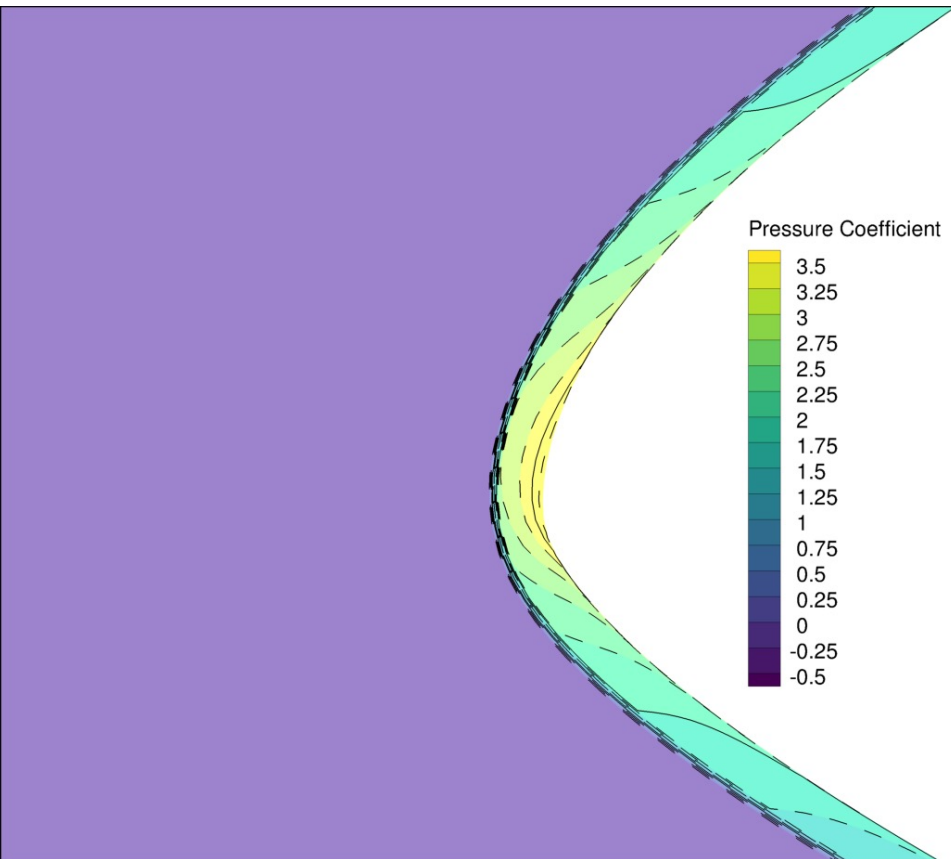
Improved SSOR, 1 processor



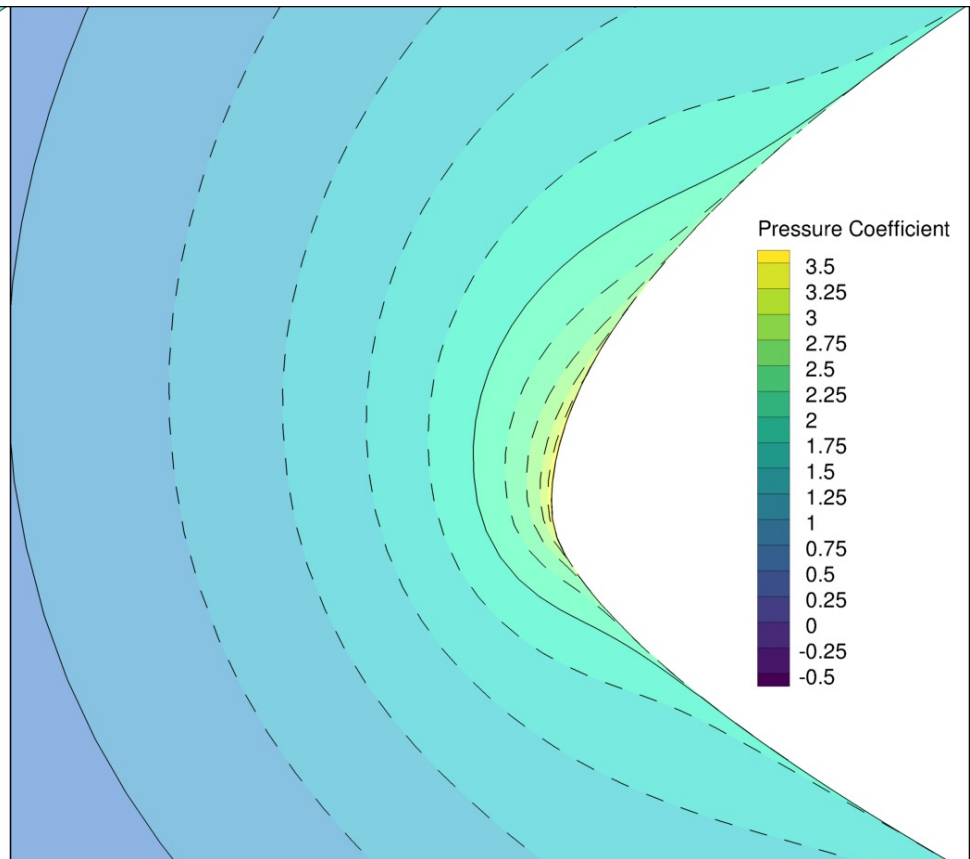
Impulsive start, one nonlinear step

Background

Improved SSOR, 8 processors



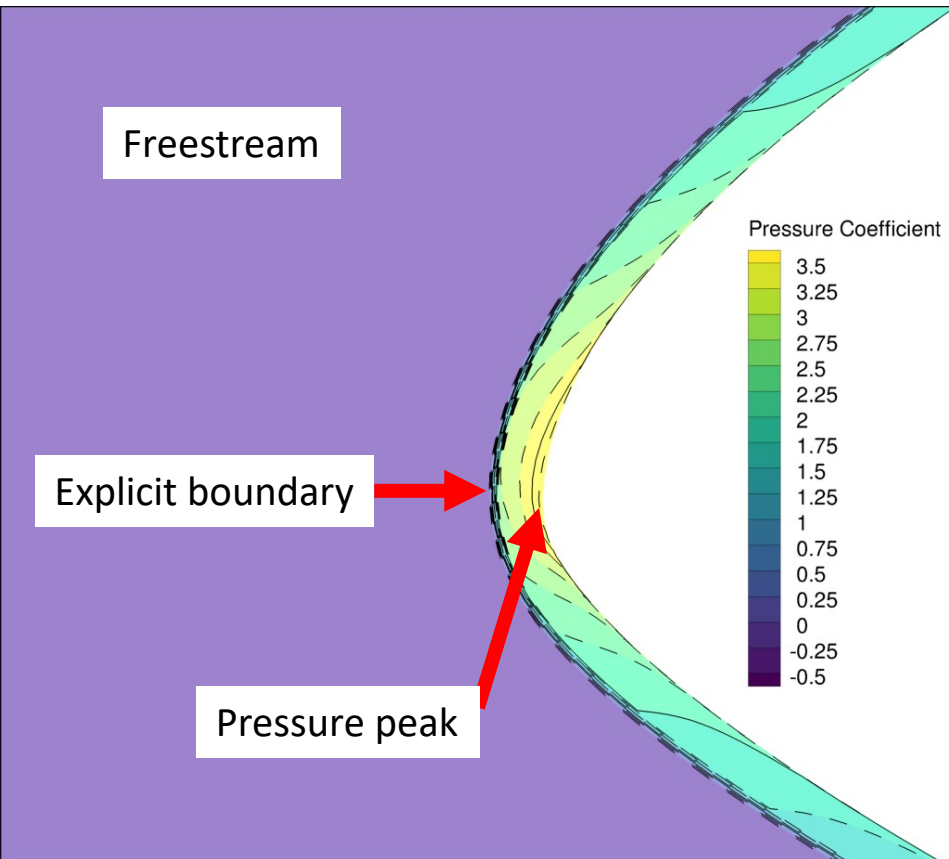
Improved SSOR, 1 processor



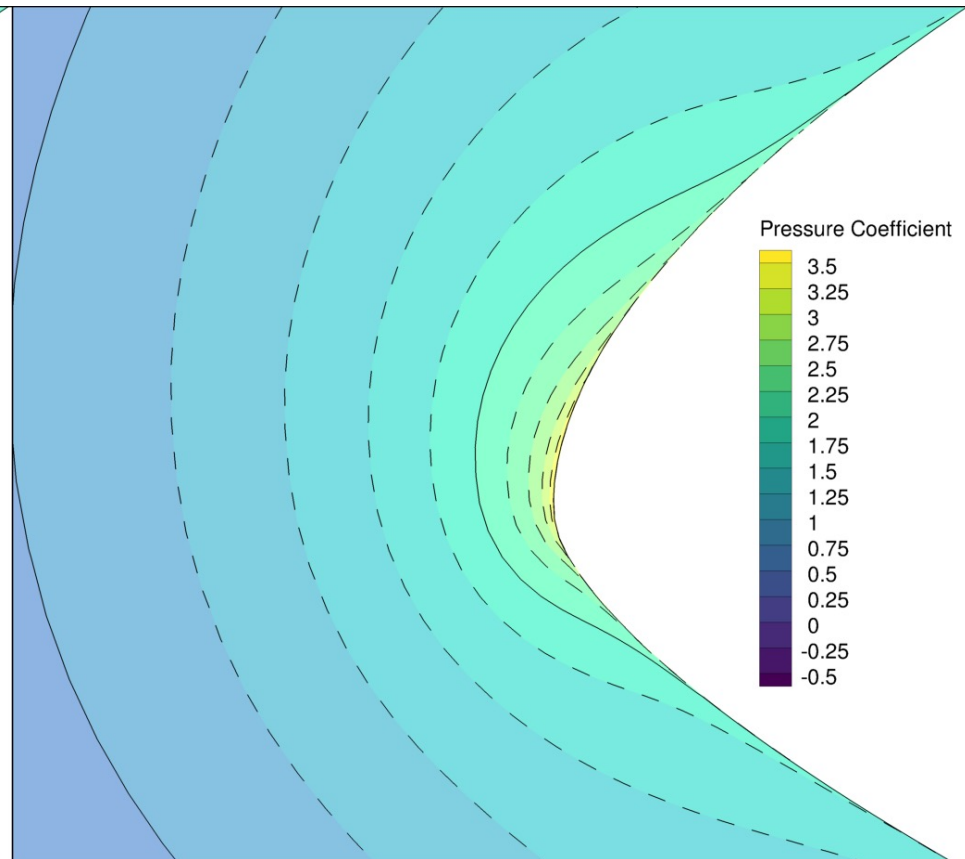
Impulsive start, one nonlinear step

Background

Improved SSOR, 8 processors



Improved SSOR, 1 processor



Impulsive start, one nonlinear step



Problems

- Explicit boundaries during the linear solve cause
 1. a large portion of the domain to not 'see' the boundaries driving the flowfield
 2. artificial constraints that can intensify the flowfield development
- As a result, the improved linearizations introduced in OVERFLOW 2.3 can exacerbate potential instabilities
- These problems could potentially be alleviated by performing data exchange during the linear solve



Parallel SSOR

- Because the SSOR scheme is a sweeping algorithm, there is an opportunity to interchange data between sweeps
 - See data-parallel algorithms such as those discussed in the paper
- The basic PSSOR algorithm for solving $Ax = b$
 1. Each grid block forms their A , b , and assumes $x_0 = 0$
 2. At the start of each symmetric sweep, all the grids do a Chimera exchange of x_{n-1} , i.e., receivers get updated linear solution values from donors
 3. Perform a single symmetric sweep
 4. Return to Step 2 until EXIT



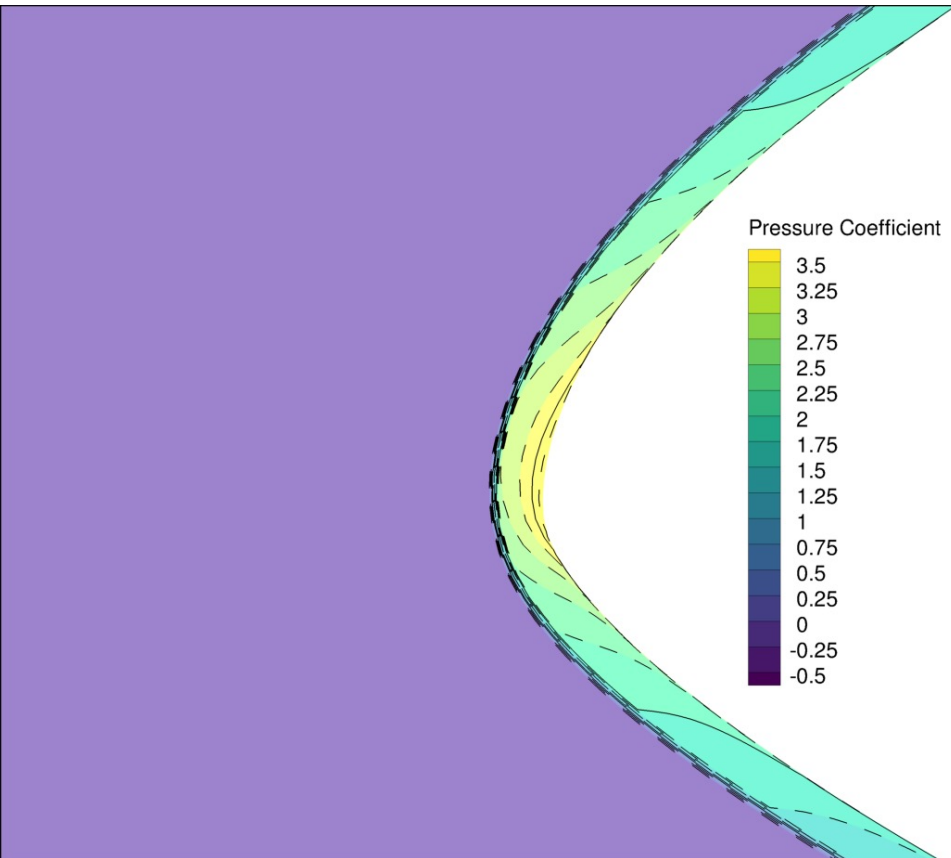
Parallel SSOR

- Since OVERFLOW is a purely overset grid code, the PSSOR algorithm is implemented with both additive and multiplicative Schwarz method options
 - Additive: receiver points only receive updated solution values (x) from donors
 - Multiplicative: receiver points that support a first order linearization will receive RHS (b) and updated solution values (x) from donors
- These changes required the addition of several synchronization points within OVERFLOW that should be mostly transparent to users
 - Wall functions have the only noticeable change, regardless of if the PSSOR scheme is used

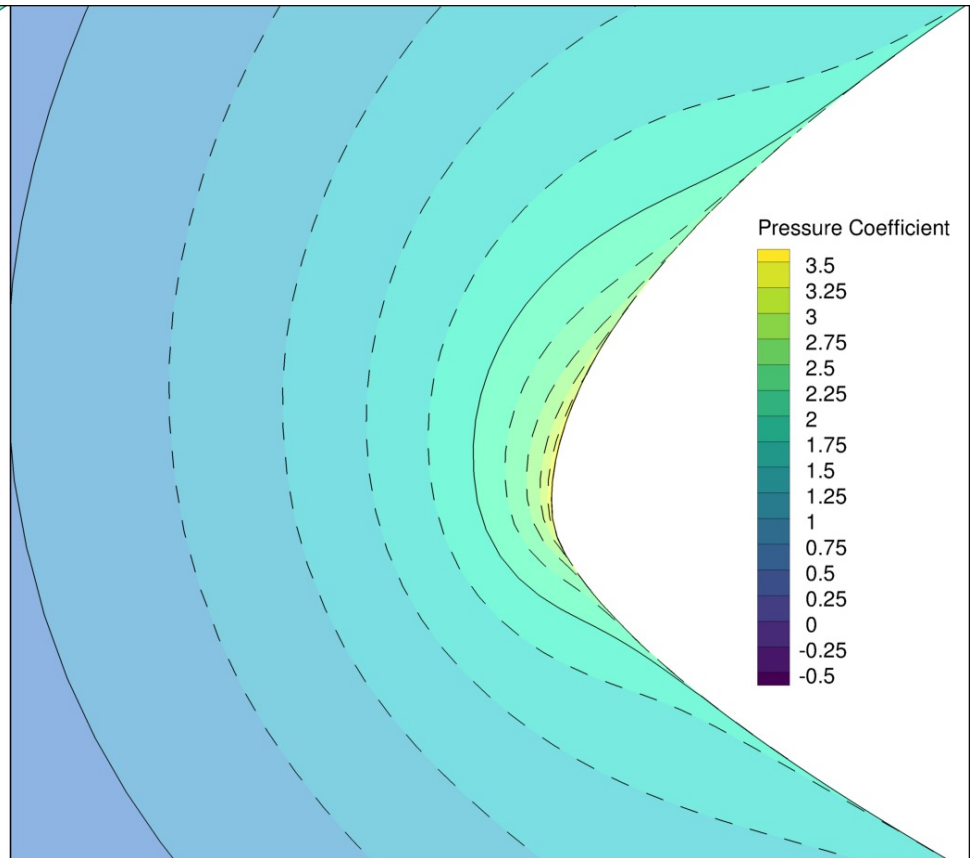


SSOR Comparison

Improved SSOR, 8 processors



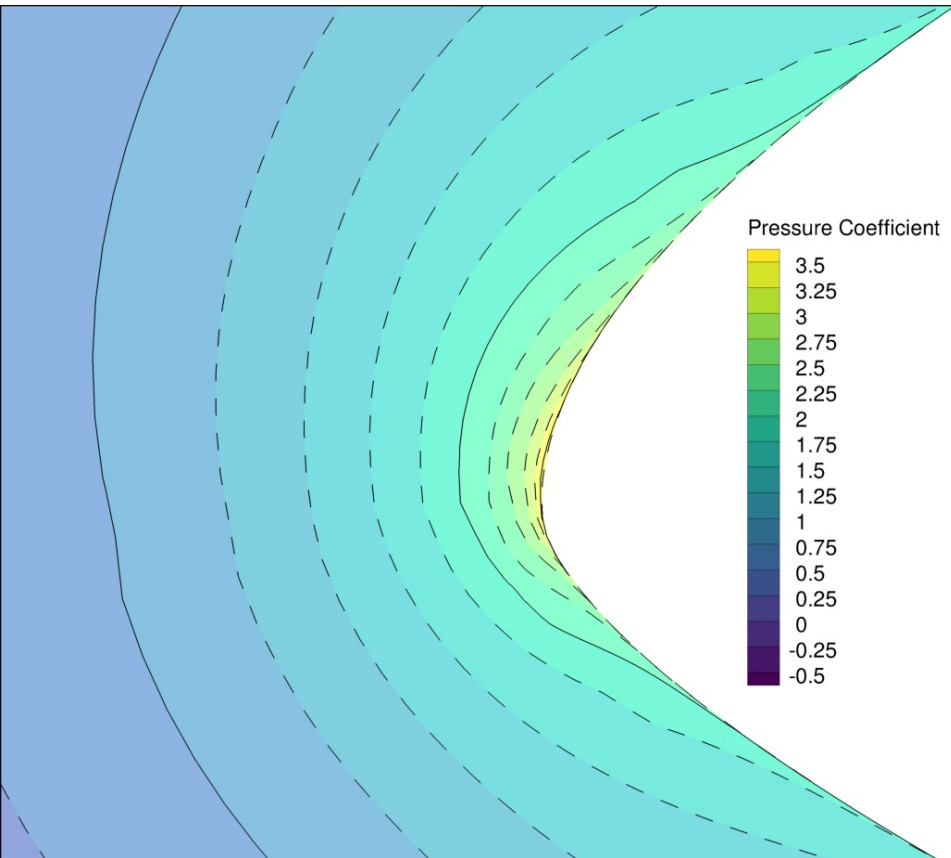
Improved SSOR, 1 processor



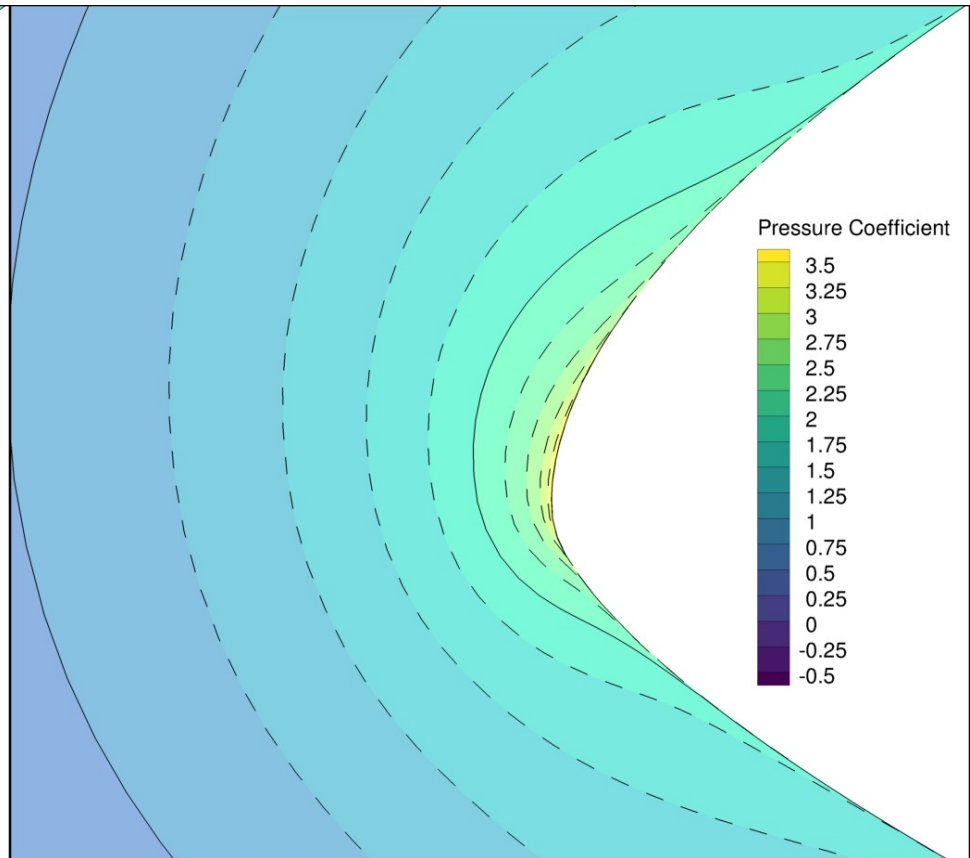
Impulsive start, one nonlinear step

SSOR Comparison

Parallel SSOR, 8 processors



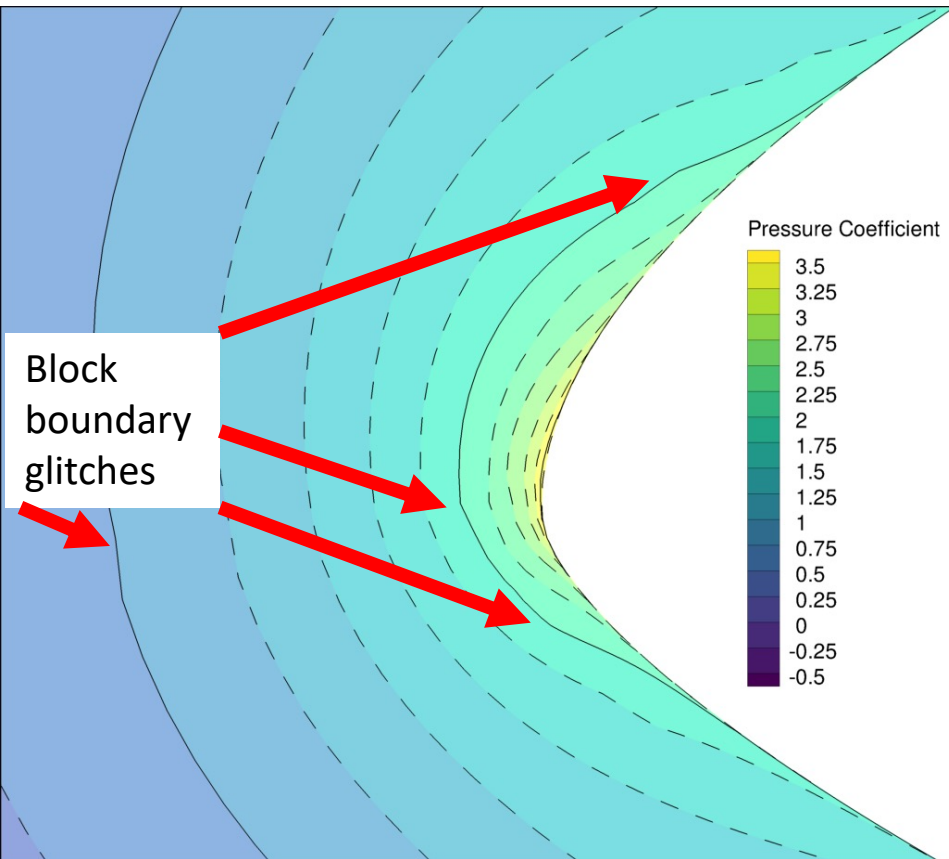
Improved SSOR, 1 processor



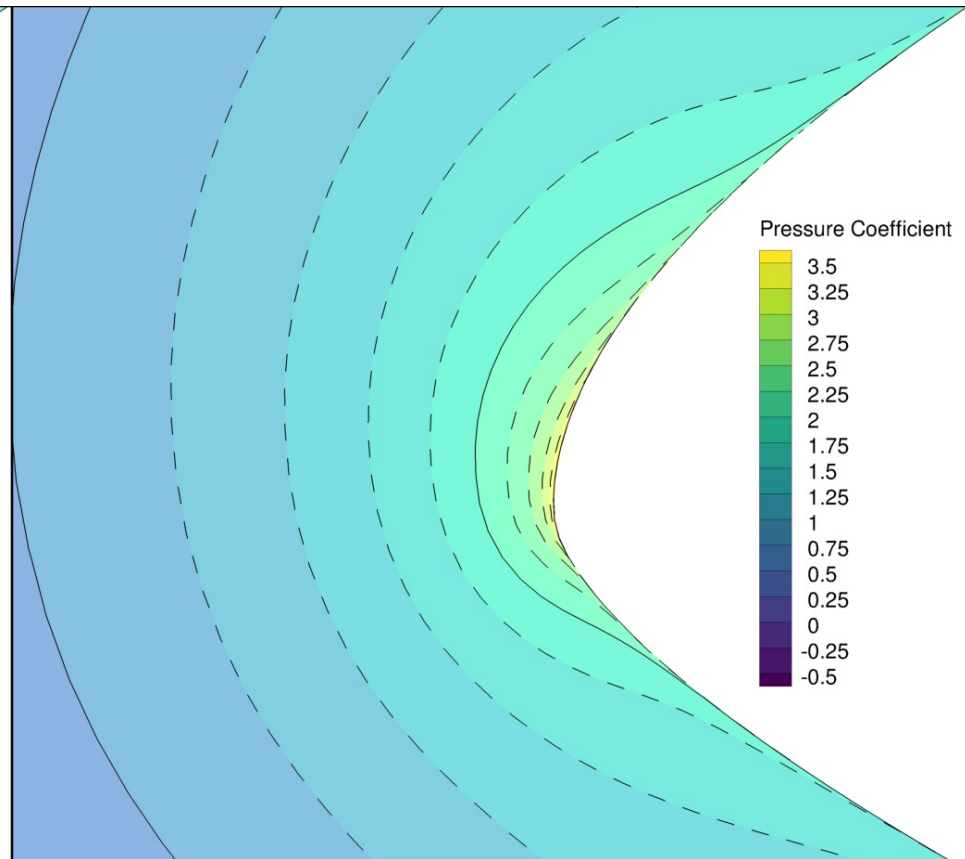
Impulsive start, one nonlinear step

SSOR Comparison

Parallel SSOR, 8 processors

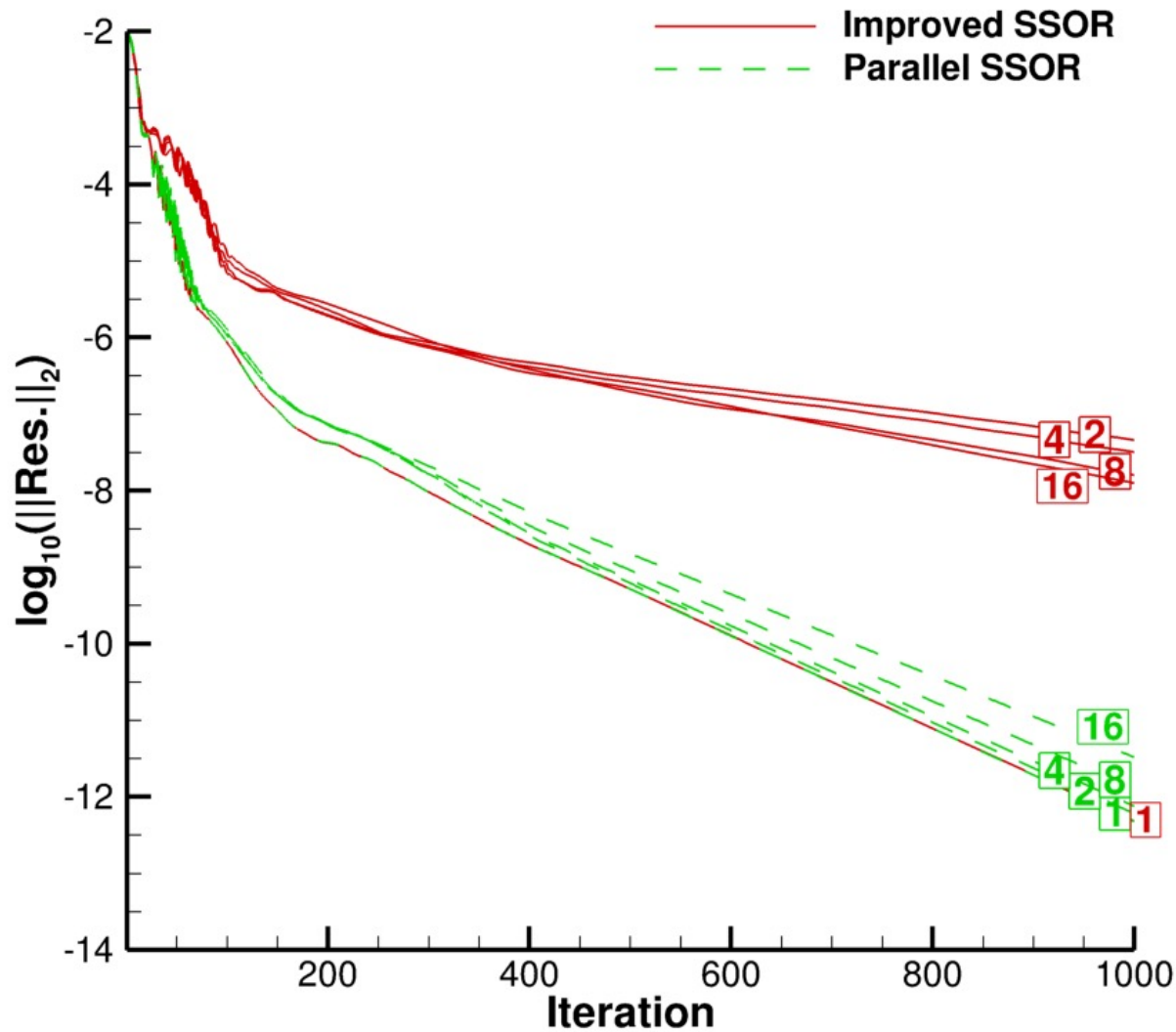


Improved SSOR, 1 processor



Impulsive start, one nonlinear step

S809 Comparison (C-grid)



Further Examples

- Please see the paper for further discussion and comparisons with AIAA 2020-1045
 - Further S809 examples
 - Pitching NACA 0015
 - 3rd Drag Prediction Workshop
 - 6th Drag Prediction Workshop
- To summarize, using the PSSOR scheme resulted in faster time-to-solution
 - Do review the DPW-6 case for an example of convergence interaction between subdomains



Conclusions

- The PSSOR scheme appears to improve initial setup behavior and greatly improves nonlinear convergence for grids split for parallel runs
 - ‘Unfortunate’ grid splitting, coarsening, or solution transients can still occur!
- Improves robustness across different numbers of MPI ranks, i.e., less random failures due to grid splitting
- This work has been implemented by rearranging the internal structure of OVERFLOW to allow for a non-block-Jacobi linear solve that should be mostly transparent to users
 - Extra memory is required by the PSSOR scheme due to storage needs while cycling through grids



Questions?

- This work was funded by the Revolutionary Vertical Lift Technology project of the Advanced Air Vehicles Program
- Special thanks to Robert ‘Bobby’ Nichols for introducing this topic after SciTech 2020
- Special thanks to Brian Allan (LaRC) for testing

