An Overset Generalized Minimal Residual Method for the Multi-Solver Paradigm

Dylan Jude*, Dr. James Baeder†
Department of Aerospace, University of Maryland

INTRODUCTION

Existing Overset Frameworks

- The Helios (HPCMP CREATE™-AV) and Mercury (University of Maryland) CFD frameworks use a multi-solver paradigm with overset meshes to model complex aerodynamic flows.
- Each solver computes the flow solution for an individual mesh at a given timestep and the solution is then exchanged at the fringe points to create a continuous global solution.
- GMRES is a linear solver often used in CFD with an implicit Euler temporal discretization.

Issues with Current Framework

- Exchange of information across mesh boundaries is done in an explicit fashion at every timestep. This can result in stalled convergence.

Desired Overset-GMRES (O-GMRES) Framework

- Instead of solving GMRES independently within each mesh, create a single overset system that considers the interpolation weights between meshes.
- Framework should be non-intrusive to existing flow solvers within Helios, fitting into the existing python interface.

$$\begin{bmatrix}[A_1] & 0 \\ 0 & [A_2]\end{bmatrix}\begin{bmatrix}[q_1] \\ [q_2]\end{bmatrix} = \begin{bmatrix}[S_1] \\ [S_2]\end{bmatrix} \qquad \begin{bmatrix}[A_1] & [B_{12}] \\ [B_{21}] & [A_2]\end{bmatrix}\begin{bmatrix}[q_1] \\ [q_2]\end{bmatrix} = \begin{bmatrix}[S_1] \\ [S_2]\end{bmatrix}$$

Meshes treated independently — Off-diagonal fringe cells considered

METHODOLOGY

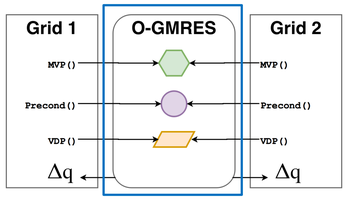
O-GMRES Algorithm

- Can be broken down into three primary (expensive) components: a **Matrix-Vector Product**, **Preconditioner**, and **Vector Dot-product.**
- Overset communication must be added within the O-GMRES algorithm. Communication routines highlighted in the algorithm on the right.
- Right preconditioning was primarily used however left-preconditioning also implemented.
- Resulting system considers residuals from fringe points within the O-GMRES algorithm.

Overset Connectivity

- The Topology Independent Overset Grid Assembler (TIOGA) was used for overset grid communication.
- Allows exchange of arbitrary vector across grids, not just solution vector. Required for exchange of residuals in O-GMRES.
- Wrapped in Python for this research.

O-GMRES module written in Python



O-GMRES Algorithm

Algorithm Right-Preconditioned O-GMRES Algorithm

1: Compute $r_o = b - Ax_o$, $\beta := ||r_o||_2$, and $v_1 := r_o/\beta$
2: **for** $j = 1, 2, ..., m$ **do**
3: OVERSET_EXCHANGE(v_j)
4: Compute $r_j :=$ PRECONDITION(v_j) ▷ Preconditioner
5: Compute $w_j := Ar_j$ ▷ Matrix-Vector Product
6: **for** $i = 1, ..., j$ **do**
7: $h_{ij} := (w_j, v_i)$ ▷ Vector Dot Product
8: OVERSET_SUM(h_{ij})
9: $w_j := w_j - h_{ij}v_i$
10: **end for**
11: $h_{j+1,j} = ||w_j||_2$
12: **if** $h_{j+1,j} == 0$ **then**
13: $m := j$
14: **goto** 18
15: **end if**
16: $v_{j+1} = w_j/h_{j+1,j}$
17: **end for**
18: Define the $(m+1) \times (m)$ Hessenberg matrix $\bar{H} = \{h_{ij}\}_{1 \le i \le m+1, 1 \le j \le m}$
19: Compute y_m, the minimum of $||\beta e_1 - \bar{H}_m y||_2$ and $r_m = V_m y_m$
20: OVERSET_EXCHANGE(r_m)
21: Compute $x_m = x_o +$ PRECONDITION(r_m)

CFD Flow Solvers

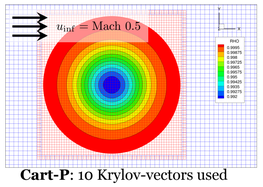
- **Cart-P**, a research code similar to samCART (in Helios), is a background Navier-Stokes flow solver for structured, Cartesian meshes.
- **mStrand** (in Helios) is a near-body Navier-Stokes solvers for strand meshes.
- **GARFIELD** (University of Maryland) is a Navier-Stokes solver for curvilinear grids.
  - GARFIELD runs on Graphics Processing Units (GPUs) to accelerate computation.
- Cart-P, mStrand and GARFIELD are wrapped in Python using F2PY or the Python-C API
- Only minor modifications to each flow solver were required to expose required functions from existing GMRES algorithm in each solver.

Resulting Python Framework

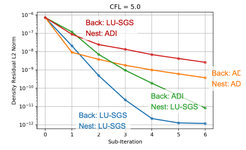
- Developed Python Framework is independent of any flow solver.
- Integrated into the Mercury framework (University of Maryland Overset CFD Framework)

RESULTS

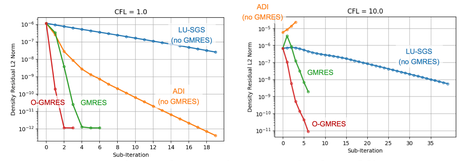
Vortex Convection



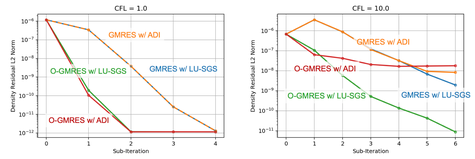
Cart-P: 10 Krylov-vectors used



Results of using different preconditioners in the fine (nested) and coarse (back- ground) meshes.

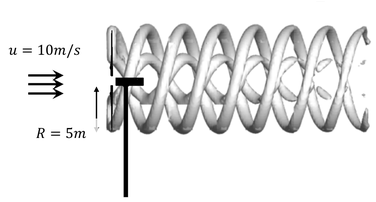


Comparison of approximate methods and linear solvers at low-CFL and moderate-CFL.

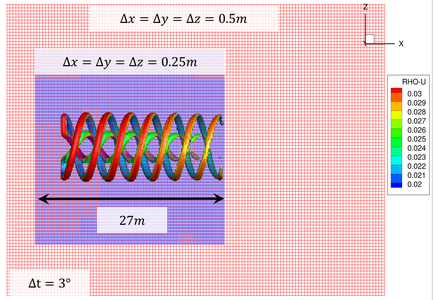


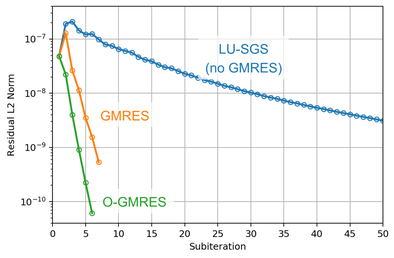
Comparison of different preconditioners for GMRES and O-GMRES for low-CFL and moderate-CFL.

NREL Phase VI Wake Capture



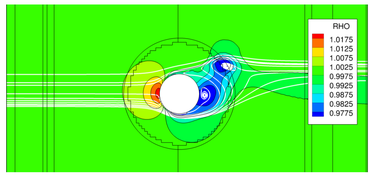
Rotor blades modeled as an actuator line. Iso-surface of vorticity shown. 5 Krylov vectors used.

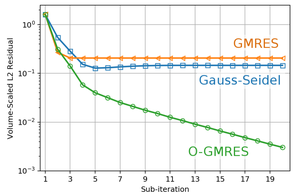


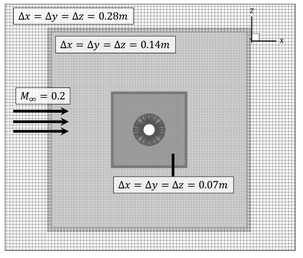


- **Cart-P** used for all meshes.
- LU-SGS and GMRES increase initially but not O-GMRES.
- O-GMRES also converges 1 more order than GMRES with 1-less subiteration.
- Convergence of LU-SGS beginning to stall with more iterations.

Multi-Solver Flow Over a Sphere



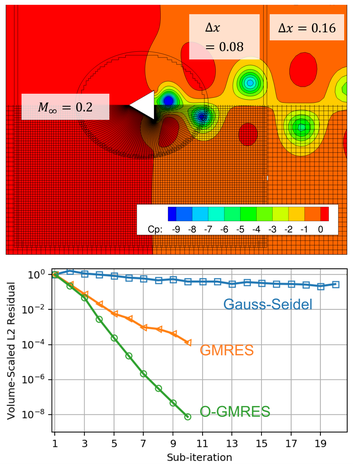




- **Cart-P** used for background, **mStrand** used for sphere.
- Different preconditioners available in sphere and background meshes.
- Gauss-Seidel refers to LU-SGS in background, Colored Line-Gauss-Seidel in sphere mesh.
- O-GMRES does not stall like GMRES or Gauss-Seidel methods

RESULTS

Flow Over a Wedge



- **GARFIELD** used for wedge and background meshes.
- Wedge side has unit length and sharp corners to induce shedding.
- Inviscid case with $M_\infty = 0.2$
- Case initialized to physical solution and then restarted using:
  - Red-Black Line Gauss Seidel for wedge
  - Diagonalized ADI for background
- A timestep corresponding to a CFL of 3 in the nested Cartesian mesh was used.
- For 10 sub-iterations, O-GMRES converges 4 more orders than GMRES.
- The Gauss-Seidel update with 20 subiterations did not reach 1 order of convergence.
- GARFIELD is a GPU solver with all exchange of data between meshes conducted on the GPU.

CONCLUSIONS

1. A study of different combinations of approximate implicit methods as preconditioners to GMRES and O-GMRES for vortex convection showed LU-SGS performed generally better than ADI as a preconditioner. Using different preconditioners between meshes also results in decreased convergence compared to the single-preconditioner case. For all cases O-GMRES out-performed GMRES.
2. Wake-capture of an NREL Phase VI wind turbine highlighted the usefulness of linear solvers where approximate methods like LU-SGS would take hundreds of iterations to converge three orders. Furthermore compared with regular GMRES, O-GMRES con- verged monotonically and achieved one extra convergence order with one less subiteration.
3. Application of O-GMRES to mStrand and Cart-P for flow over a laminar sphere showed the O-GMRES framework can easily be extended to work with existing solvers with only minimal required changes. Time-accurate convergence showed O-GMRES converging while GMRES and Gauss-Seidel stalled.
4. Application of O-GMRES to GARFIELD for flow over a wedge showed the O-GMRES framework can be used with GPU codes as well as CPU codes. Time-accurate convergence results showed O-GMRES converging much better than GMRES and Gauss-Seidel methods.
5. The computational cost of O-GMRES is hardware dependent however timing results indicate approximately 20% increase in computation time.

FUTURE DIRECTIONS

- The current O-GMRES implementation assumes Newton timestepping is used in all solvers. Future work will investigate using dual-timestepping to relax temporal source terms.
- More challenging cases involving grid motion, interactional aerodynamics, and more solver combinations will be run and presented at AIAA SciTech in January, 2019.
- Future results should also consider effects of the turbulence model, which typically has a negative effect on convergence. The turbulence model can be added to the GMRES algorithm or solved separately using a loose-coupling approach.
- Since O-GMRES is most beneficial in areas of high convection of structures between meshes, the framework could potentially be used for only a subset of meshes and not the entire mesh system.

ACKNOWLEDGEMENTS

Preliminary research for this work was conducted during an internship with the Army Aviation Development Directorate (ADD-Ames) funded by the HPCMP Internship Program and mentored by Andrew Wissink. Computational resources were used with the help of the Army Aviation Development Directorate (ADD-Ames) and Rajneesh Singh (ARL).

*PhD Candidate, University of Maryland †Professor, University of Maryland