Higher Order Multigrid Algorithms for a 2D and 3D RANS-$k\omega$ DG-Solver

Marcel Wallraff, Tobias Leicht

DLR Braunschweig (AS - C$^2$A$^2$S$^2$E)
Multigrid algorithms
DG discretization

*Basis functions*

- non-parametric ortho-normal basis functions
- directly formulated in physical space
- also referred to as Taylor-DG
- need to be evaluated for each mesh element

*RANS-*\(k\omega\) *equations*

- \(k\omega\) turbulence model
- second scheme of Bassi and Rebay (BR2) for the viscous terms
- Roe flux as a convective flux, based on an eigen-decomposition of the full jacobian
Non-linear multigrid method

nested hierarchy of linear spaces

\[ \mathbf{V}_{l_{\text{min}}} \subset \mathbf{V}_{l_{\text{min}}+1} \subset \cdots \subset \mathbf{V}_{l_{\text{max}}-1} \subset \mathbf{V}_{l_{\text{max}}} \]

\[ \mathbb{R}^{n_{l_{\text{min}}}} \subset \mathbb{R}^{n_{l_{\text{min}}+1}} \subset \cdots \subset \mathbb{R}^{n_{l_{\text{max}}}} \]

intergrid transfer operators:

\[ \rightarrow \text{ prolongation: natural injection } \mathcal{I}_{l-1}^l : \mathbb{R}^{n_l-1} \rightarrow \mathbb{R}^{n_l} \]

\[ \rightarrow \text{ canonical restriction operator } \mathcal{I}_{l-1}^l := \left( \mathcal{I}_{l-1}^l \right)^T \]

non-linear multigrid algorithm also requires:

\[ \rightarrow \text{ restricted nonlinear state vector: orthogonal } L^2\text{-projection } \hat{\mathcal{I}}_{l-1}^{-1} \text{ on the space } \mathbf{V}_{l-1} \]
Non-linear multigrid method

Let the non-linear problem to be solved on the fine level \( l_{\text{max}} \) be given by

\[
L_{l_{\text{max}}} (u_{l_{\text{max}}}) = f_{l_{\text{max}}},
\]

\( \Rightarrow \)

\( \Rightarrow \)

\( \Rightarrow \)

restrict solution approximation \( u_{l-1} := \hat{l}_{l}^{-1} u_{l} \)

compute forcing function for the coarse level:

\[
f_{l-1} \leftarrow f_{l-1} + \hat{l}_{l}^{-1} (f_{l} - L_{l}(u_{l})) - \left( f_{l-1} - L_{l-1}(u_{0}^{l}) \right)
\]

Galerkin-transfer for the Jacobian: \( R_{l-1} = \hat{l}_{l}^{-1} R_{l} \hat{l}_{l-1}^{-1} \)
Non-linear smoother / solver

**smoother / solver**

- linearized Backward-Euler
  - **Solve** \[ \left( (\alpha_i \Delta t)^{-1} M + R_l \right) (u_{l,i} - u_{l,i-1}) = \left[ f_l - L_l(u_{l,i-1}) \right], \]
  - where \( R_l \) is the fully implicit Jacobian matrix and \( M \) is the mass matrix. In addition to that \( u_{l,j} \) is a state vector, with \( u_{l,j} \in V_l \forall j \in \mathbb{N}. \)

- local pseudo-time steps, adaptive CFL number
Linear smoother / solver

- Krylov method as linear solver (GMRES method)
- line-Jacobi as preconditioner / smoother
  - let $\mathcal{L}_{l,k}(u_{l,k}) = f_{l,k}$ the underlying linear problem on line $k$,
  - solve $\delta u_{l,k,i} := u_{l,k,i} - u_{l,k,i-1} = R_{l,k}^{-1}(f_{l,k} - \mathcal{L}_{l,k}u_{l,k,i-1})$
  - set $u_{l,k,i} := u_{l,k,i-1} + \delta u_{l,k,i}$,
  - where $R_{l,k}^{-1}$ is the inverse of the Jacobian matrix computed one line $k$ in the mesh
Relaxation scheme

Jacobian / system matrix structure

matrix blocks
- element diagonal
- line neighbor
- off-line off-diagonal
Numerical algorithms

possible solver choices

- single grid Backward-Euler
- start up strategy in mesh or order sequencing for improved initial conditions
- linear MG as preconditioner
- non-linear MG to accelerate process in pseudo-time
- non-linear MG with linear MG on each level
Numerical parameters for the non-linear problems

*non-linear multigrid*

- only V-cycles will be presented
- one pre- and post-smoothing iteration on each level
- one smoothing iteration on the lowest level
- a linearized Backward-Euler scheme as smoother
- using an SER time stepping scheme for the Backward-Euler
- Galerkin-transfer to obtain the Jacobian on the lower levels
Numerical parameters for the linear problems

parameters for solving the resulting linear problems on every level from the Backward-Euler linearization

- GMRES method with a fixed number of max steps on every level
- linear multigrid as a preconditioner for the GMRES method
- one V-cycle is done for preconditioning
- no pre-smoothing iterations
- four post-smoothing iterations
- four smoothing iterations on the lowest level
- line-Jacobi scheme as smoother
- Galerkin-transfer to obtain the lower level matrices
L1T2 high-lift configuration

- flow conditions:
  - Mach: 0.197
  - Reynolds number: 3,520,000
  - $\alpha = 20.18^\circ$
- testcase from EC funded ADIGMA project
- computations will be shown on an unstructured mesh with 23824 mixed elements
- $p = 2$ solution is desired for all computations
L1T2 high-lift configuration

mixed-element mesh (CENAERO via GMSH)
third order
142 944 DoF
three levels
L1T2 high-lift configuration

run time comparison: $h$-MG

normalized CPU time

density residual

- $SG$
- $h$-sequence
- $h$-NMG
- $h$-LMG
- $h$-NMG + LMG
L1T2 high-lift configuration

run time comparison: $p$-MG

![Graph showing run time comparison for different configurations: SG, $p$-sequence, $p$-NMG, $p$-LMG, $p$-NMG + LMG with normalized CPU time on the x-axis and density residual on the y-axis.](image-url)
VFE-2 Delta-Wing with rounded leading edge

- flow conditions:
  - Mach: 0.4
  - Reynolds number: 3,000,000
  - $\alpha = 13.3^\circ$, $\beta = 0^\circ$

- testcase from EC funded IDIHOM project

- computations will be shown on a structured mesh with 13816 elements

- $p = 2$ solution is desired for all computations
VFE-2 Delta-Wing with rounded leading edge

- original FV mesh with 884,224 elements was two times agglomerated
- resulting in a higher order mesh with 13,816 curved elements

original mesh with straight faces

curved mesh with faces represented by polynomials of order 4
VFE-2 Delta-Wing with rounded leading edge

![Graph showing density residual vs. top level non-linear iterations for different methods: h-sequence, h-LMG, h-NMG, p-sequenc, p-LMG, p-NMG, p-NMG + LMG.](image)
VFE-2 Delta-Wing with rounded leading edge

![Graph showing the density residual over top level non-linear iterations for different methods: $h$-NMG + LMG, $h$-NMG + LMG, $p$-NMG + LMG, $p$-NMG + LMG.](image-url)
VFE-2 Delta-Wing with rounded leading edge

The *blue* computation on the mesh with 13816 elements and the *red* computation on an adjoint refined mesh with 23877 elements.
Thanks for your attention!