14th World Congress on Computational Mechanics (WCCM)
ECCOMAS Congress 2020)
Virtual Congress: 11-15 January 2021
F. Chinesta, R. Abgrall, O. Allix and M. Kaliske (Eds)

AN FINITE VOLUME BASED MULTIGRID PRECONDITIONER FOR DG-SEM FOR CONVECTION-DIFFUSION

Johannes KASIMIR[*,1], **Lea M. VERSBACH**[*,2], **Philipp BIRKEN**[*,3], **Gregor J. GASSNER**[4] **AND Robert KLÖFKORN**[*,5]

[*] Lund University
Centre for Mathematical Sciences, Box 118, 221 00 Lund, Sweden

[1] johannes.kasimir.418@student.lu.se

[2] lea_miko.versbach@math.lu.se, http://www.maths.lu.se/staff/lea-miko-versbach/

[3] philipp.birken@math.lu.se, http://www.maths.lu.se/staff/philipp-birken/

[4] University of Cologne
Division of Mathematics/Center for Data and Simulation Science, Weyertal 86 90,50931 Köln,
Germany, ggassner@math.uni-koeln.de, https://www.mi.uni-koeln.de/NumSim/gregor-gassner/

[5] robertk@math.lu.se, http://www.maths.lu.se/staff/robert-kloefkorn/

Key words: Discontinuous Galerkin, Finite Volume, Multigrid Preconditioners, Compressible Flow

Abstract. The goal of our research is the construction of efficient Jacobian-free preconditioners for high order Discontinuous Galerkin (DG) discretizations with implicit time integration. We are motivated by three-dimensional unsteady compressible flow applications, which often result in large stiff systems. Implicit time integrators overcome the impact upon restrictive CFL conditions on explicit ones but leave the problem to solve huge nonlinear systems. In this paper we consider a multigrid preconditioning strategy for Jacobian-free Newton-Krylov (JFNK) methods for the solution of algebraic equation systems arising from implicit Discontinuous Galerkin (DG) discretizations. The preconditioner is defined by an auxiliary first order Finite Volume (FV) discretization that refines the original DG mesh, but can still be implemented algebraically. Different options exist to define the grid transfer between DG and FV. We suggest an ad hoc assignment of the unknowns as well as L_2 projections. We present new numerical results for the two-dimensional convection-diffusion equation in combination with the different transfer options, which demonstrate the quality and efficiency of the suggested preconditioner with regards to convergence speed up and CPU time. The suggested L_2 projection from this paper result in the best convergence speed up.

1 INTRODUCTION

In our research we are motivated by three-dimensional unsteady compressible flow applications. Our goal is the construction of efficient Jacobian-free preconditioners for high order Discontinuous Galerkin (DG) discretizations with implicit time integration. High order methods such as DG or Flux Reconstruction discretizations offer great potential for Large Eddy Simulation (LES) of turbulent flows with complex geometries, e.g. jet engines. The idea of these discretizations is an element-wise polynomial approximation, with possible discontinuities across element interfaces [1, 2]. The degrees of freedom are coupled across faces and inside each element, resulting in dense element-wise computation. Thus, these methods are very well suited for high performance computing [3, 4]. We consider a specific DG variant, the DG Spectral Element Method (DG-SEM), see e.g. [5]. This discretization is based on a Lagrange type (nodal) basis with Legendre-Gauss-Lobatto (LGL) quadrature nodes, with collocation of the discrete integration. Moreover, the same Lagrange basis is used for ansatz and test space. These choices yield DG operators that satisfy the summation-by-parts (SBP) property [6], which is the discrete analogue to integration-by-parts. This property is key to construct methods that are discretely entropy stable and/or kinetic energy preserving.

Discontinuous Galerkin discretization in space results in a big system of ODEs. In a lot of challenging compressible turbulent flow applications as e.g. the design of jet engines, airplanes and windturbines, these systems are stiff due to geometry features and thin boundary layers. It is therefore of advantage to use implicit time integrators to overcome the deficiency of explicit ones with restrictive CFL conditions. However, this causes challenges in terms of CPU time and memory consumption. In [7], Vincent and Jameson mention that solvers for linear and nonlinear equation systems are severely lacking for 3D DG applications, and are therefore one of four major obstacles that need to be solved if high order methods are to be widely adopted by e.g. industry. Full-Approximation multigrid (FAS-Multigrid) schemes and preconditioned Jacobian-Free Newton-Krylov methods (JFNK) combined with multigrid preconditioners are candidates for such efficients solvers [8, 9]. We focus on the latter one and the construction of efficient preconditioners. DG systems have a block structure with weakly coupled blocks at the faces. The blocks can be of very large size depending on the order of the DG method: the number of unknowns per element increases dramatically with increasing polynomial degree and dimension, leading to large dense Jacobian blocks [10, 11]. While for Euler or Navier-Stokes equations the block size of a Finite Volume method is $2+d$ with dimension d, it is $(d+2)\,(p+1)^d$ for DG-SEM with p-th degree polynomials on quadrilateral grids. For degree 2 polynomials in 3D this results in dense blocks of size 135×135. Thus, the favorable memory consumption of the JFNK approach is obsolete if the preconditioner requires the storage of the DG system Jacobian.

In this article we extend the idea for the construction of a well-performing low storage preconditioner for the JFNK approach, i.e. a Jacobian-free preconditioner as suggested in [12, 13], to the two-dimensional case and consider a convection-diffusion problem. This extension is straightforward using a tensor product ansatz for the DG-SEM discretization. The main ingredient is the construction of a simplified replacement operator. While it is possible to choose a different polynomial order in the element to generate a replacement operator as in [14, 10], we aim to retain the number of DOFs in our replacement operator by introducing subcells in each element, namely $p+1$ in each spatial direction. A related approach was proposed in [15] for spectral difference (SD) methods. The simplest replacement operator is a first order Finite Volume (FV) discretization defined on this subcell grid. To do so, the nodal DOFs from the DG method need to be reinterpreted as unknowns for the FV method. In extension to previous work, we consider different transfer operators between the DG-SEM and FV discretization based on L_2 projections.

We measure the quality and efficiency of the suggested preconditioners by considering the convergence rate of the Krylov subsolver as well as the CPU time. Moreover, the FV discretization itself can be interpreted as a reference preconditioner. This is motivated by the fact that a well constructed multigrid method should approximate the FV method as well as possible. All results presented in this paper are produced using the Python bindings in the Distributed and Unified Numerics Environment (DUNE) [16].

This is a modular toolbox for solving partial differential equations with grid-based methods, ensuring efficiency in scientific computations and supporting high-performance computing applications. Numerical methods can be developed and tested in Python, before they are transferred to C++ to improve performance efficiency.

In the remainder of this paper, we first describe the here considered problem, the two-dimensional convection diffusion-equation. We then present the DG-SEM discretization and the FV subcell discretization. In section 3 we describe how to construct an agglomeration multigrid preconditioner and present the different transfer operators between DG and FV. In the last part of the paper, we show numerical experiments to validate the approach and draw our conclusions.

2 PROBLEM DESCRIPTION AND DISCRETIZATION

In this paper we consider the two-dimensional linear convection-diffusion equation

$$\begin{gathered}\mathbf{u}_t + \mathbf{b}\cdot\nabla\mathbf{u} - \varepsilon\Delta\mathbf{u} = \mathbf{g}(\mathbf{u}), \\ \mathbf{u}_y|_{\Omega_{SN}} = \mathbf{0},\ \mathbf{u}|_{\partial\Omega_{WE}} = \mathbf{0}, \\ \mathbf{u}(\mathbf{X},0) = \exp(-10\|\mathbf{x}\|^2),\end{gathered} \tag{1}$$

with $(\mathbf{X},t)\in\Omega\times[0,T]$, $\Omega=[-1,1]^2$, $\mathbf{b}=\frac{1}{2}(\sqrt{3},1)$ and $\varepsilon\in\mathbb{R}^+$.

2.1 Discontinuous Galerkin Spectral Element Method

We discretize the spatial components of (1) using a discontinuous Galerkin method and rewrite problem (1) as

$$\mathbf{u}_t + \nabla\cdot\mathbf{f}(\mathbf{u},\nabla\mathbf{u}) = \mathbf{0},\ (\mathbf{X},t)\in\Omega\times[0,T] \text{ with } \Omega\subset\mathbb{R}^d. \tag{2}$$

To derive the discretization we consider the weak form of the conservation law

$$\int_\Omega \mathbf{u}_t\psi\, d\Omega + \int_\Omega \nabla\cdot\mathbf{f}\psi\, d\Omega = \mathbf{0}, \tag{3}$$

using test functions ψ from some test space. Integration by parts yields

$$\int_\Omega \mathbf{u}_t\psi\, d\Omega + \int_{\partial\Omega} \mathbf{f}\cdot\mathbf{n}\psi\, ds - \int_\Omega \mathbf{f}\cdot\nabla\psi\, d\Omega = \mathbf{0}. \tag{4}$$

In the next step, the domain Ω is subdivided into cells Ω_i. In each cell the solution $\mathbf{u}$ and the flux function $\mathbf{f}(\mathbf{u},\nabla\mathbf{u})$ are approximated by a polynomial. In the following we assume that $\Omega\subset\mathbb{R}^2$, which simplifies the notation. Moreover, we map all cells Ω_i onto the unit cell $\hat{\Omega} := [-1,1]^2$ as described in [1, 17]. Then the numerical approximation on the unit cell is given by

$$\mathbf{u}(\mathbf{x},t)|_{\hat{\Omega}} \approx \quad \mathbf{u}^P(\mathbf{x},t) = \textstyle\sum_{j=1}^{N_x}\sum_{k=1}^{N_y} \mathbf{u}_{jk}(t)\varphi_{jk}(\mathbf{x}). \tag{5}$$

The choice of the basis functions gives rise to different DG methods. We consider here the DG spectral element method (DG-SEM), where the same Lagrange basis is used for ansatz and test space. Moreover, the flux function is approximated by

$$\mathbf{f}(\mathbf{u},\nabla\mathbf{u})|_{\hat{\Omega}} \approx \mathbf{f}^P(\mathbf{u},\mathbf{x},t) = \sum_{j=1}^{N_x}\sum_{k=1}^{N_y} \mathbf{f}_{jk}(t)\varphi_{jk}(\mathbf{x}), \tag{6}$$

with polynomial basis functions $\varphi_{jk}(\mathbf{x}) = \varphi_j(x)\varphi_k(y)$ of degree p and $\mathbf{f}_{jk}(t) = \mathbf{f}(\mathbf{u}_{jk}(t), \nabla\mathbf{u}_{jk}(t))$. The global solution is then approximated by a piecewise polynomial. We only have continuity inside the elements, but not on the interfaces, i.e. we allow for discontinuous polynomial approximations. The test functions are also chosen to be polynomials of degree up to p. Then (4) becomes on the reference cell

$$J\int_{\hat{\Omega}} \mathbf{u}_t^P \varphi_{jk}\, d\hat{\Omega} + \int_{\partial\hat{\Omega}} \mathbf{f}^P \cdot \mathbf{n}\varphi_{jk}\, d\mathbf{s} - \int_{\hat{\Omega}} \mathbf{f}^P \cdot \nabla\varphi_{jk}\, d\hat{\Omega} = \mathbf{0},\ j = 1,\dots,N_x, k = 1,\dots,N_y, \tag{7}$$

with $J = \frac{\Delta x \Delta y}{4}$. As polynomial basis for the DG-SEM method we choose a Lagrange basis with a nodal basis based on Legendre-Gauss-Lobatto (LGL) nodes, which are defined on the unit cell $\hat{\Omega}$. Next we apply a Gaussian quadrature to approximate the integrals. For the DG-SEM method, interpolation and quadrature are collocated. Thus polynomials up to degree $2p-1$ can be approximated exactly.

This gives for the mass integral

$$J\int_{\hat{\Omega}} \mathbf{u}_t^P \varphi_{jk}\, d\hat{\Omega} = J\sum_{l=1}^{N_x}\sum_{m=1}^{N_y} \dot{\mathbf{u}}_{lm}(t)\int_{\hat{\Omega}} \varphi_{lm}\varphi_{jk}\, d\hat{\Omega} \approx J\dot{\mathbf{u}}_{jk}(t)\omega_j\omega_k,\ j = 1,\dots,N_x, k = 1,\dots,N_y. \tag{8}$$

Defining a vector of coefficients $\tilde{\mathbf{u}}$ and a diagonal mass matrix $\mathbf{M}$ with diagonal elements given by the quadrature weights, equation (8) can be written more compactly as $J\mathbf{M}\tilde{\mathbf{u}}$.

Applying Gaussian quadrature to the volume integral yields

$$\begin{aligned} \int_{\hat{\Omega}} \mathbf{f}^P \cdot \nabla\varphi_{jk}\, d\hat{\Omega} &= \int_{\hat{\Omega}} \sum_{l=1}^{N_x}\sum_{m=1}^{N_y} \mathbf{f}_{lm}(t)\varphi_{lm} \cdot \nabla\varphi_{jk}\, d\hat{\Omega} \\ &\approx \sum_{l=1}^{N_x} \omega_l\omega_k(\mathbf{f}_{lk})_1\varphi_j'(x_l) + \sum_{m=1}^{N_y} \omega_j\omega_m(\mathbf{f}_{jm})_2\varphi_k'(x_m),\ j = 1,\dots,N_x, k = 1,\dots,N_y. \end{aligned} \tag{9}$$

With $\tilde{\mathbf{f}}$ the vector of evaluations at the quadrature nodes and $\mathbf{S}$ such that

$$\begin{aligned} (\mathbf{S}_{jk})_1 &= \omega_l\omega_k\varphi_j'(\mathbf{x}_l),\ j = 1,\dots,N_x, k = 1,\dots,N_y, \\ (\mathbf{S}_{jk})_2 &= \omega_j\omega_m\varphi_k'(\mathbf{x}_m),\ j = 1,\dots,N_x, k = 1,\dots,N_y, \end{aligned} \tag{10}$$

equation (9) can be written compactly as $\sum_{i=1}^{2} \mathbf{S}_i(\tilde{\mathbf{f}})_i$.

The boundary integral is the sum of the four integrals on the edges of the reference cell. Then the integral can be calculated as

$$\begin{aligned} \int_{\partial\Omega_i} \mathbf{f}^P \cdot \mathbf{n}\varphi_{jk}\, d\mathbf{s} &\approx \omega_k(\mathbf{f}^P(\mathbf{u},(1,y_k),t))_1\varphi_j(1) - \omega_k(\mathbf{f}^P(\mathbf{u},(-1,y_k),t))_1\varphi_j(-1) \\ &+ \omega_j(\mathbf{f}^P(\mathbf{u},(x_j,1),t))_2\varphi_k(1) - \omega_j(\mathbf{f}^P(\mathbf{u},(x_j,-1),t))_2\varphi_k(-1), \\ &\quad j = 1,\dots,N_x, k = 1,\dots,N_y. \end{aligned} \tag{11}$$

We define the vector of function evaluations at the quadrature nodes on the surface as $\hat{\mathbf{f}}$ and $\mathbf{M}^s_{jk} = \omega_j\varphi_k(\mathbf{x}_{face})$, then equation (11) can be written compactly as $\sum_{i=1}^{4}(\mathbf{M}^s_{jk})_i(\hat{\mathbf{f}})_i$. Next, the boundary terms need to be coupled with the neighboring cells and since we allow for discontinuities at the cell boundaries for numerical solution, numerical flux functions are needed. We approximate

$$\mathbf{f}^P(\mathbf{x},t) \approx \mathbf{f}_c^N(\mathbf{u}^P(\mathbf{x},t), \hat{\mathbf{u}}^P(\mathbf{x},t); \mathbf{n}) + \mathbf{f}_d^N(\mathbf{u}^P(\mathbf{x},t), \nabla\mathbf{u}^P(\mathbf{x},t), \hat{\mathbf{u}}^P(\mathbf{x},t), \nabla\hat{\mathbf{u}}^P(\mathbf{x},t); \mathbf{n}), \tag{12}$$

where we write the numerical flux as a sum of the convective and the diffusive numerical fluxes, $\hat{\mathbf{u}}^P$ is the polynomial in the neighboring cell and $\mathbf{n}$ the normal.

In this paper we use the upwind flux for the convective numerical fluxes and an interior penalty Galerkin flux in the diffusive part, see [18]:

$$\mathbf{f}^N_{IP}(\mathbf{u}^P(\mathbf{x},t),\nabla\mathbf{u}^P(\mathbf{x},t);\hat{\mathbf{u}}^P(\mathbf{x},t),\nabla\hat{\mathbf{u}}^P(\mathbf{x},t);\mathbf{n}) := \{\{\varepsilon\nabla\mathbf{u}^P\}\} - \frac{\eta}{h}\varepsilon[\![\mathbf{u}^P]\!], \tag{13}$$

with the classic average $\{\{\cdot\}\}$ and jump $[\![\cdot]\!]$, local mesh width h and a stability coefficient η which we set

$$\eta = \begin{cases} 10 \times \text{order}^2, & \text{if order} > 0, \\ 1, & \text{else.} \end{cases} \tag{14}$$

On the reference cell we obtain a system of ordinary differential equations:

$$J\mathbf{M}\tilde{\mathbf{u}}_t + \sum_{j=1}^{4}\mathbf{M}^S_j(\mathbf{f}^N)_j - \sum_{i=1}^{2}\mathbf{S}_i(\tilde{\mathbf{f}})_j = \mathbf{0}. \tag{15}$$

Collecting the equations for all elements in one big system with vector of unknowns $\underline{\mathbf{u}}$ and applying implicit Euler time stepping gives

$$\begin{aligned} \underline{\mathbf{u}}^{n+1} - \underline{\mathbf{u}}^n - \Delta t\underline{\mathbf{G}}\underline{\mathbf{u}}^{n+1} &= \mathbf{0}, \\ \Leftrightarrow (\underline{\mathbf{I}} - \Delta t\underline{\mathbf{G}})\underline{\mathbf{u}}^{n+1} &= \underline{\mathbf{u}}^n, \end{aligned} \tag{16}$$

where $\underline{\mathbf{G}}$ is a matrix since we have a linear problem. This linear system of equations can be solved using a preconditioned Jacobian-free GMRES method. Matrix-vector products with the Jacobian are replaced by a finite difference approximation to get a Jacobian-free variant.

In the next section we describe how to construct a preconditioner for this linear system based on a multigrid (MG) method for a Finite Volume replacement operator.

3 MULTIGRID BASED PRECONDITIONER

In order to construct a preconditioner for the linear system (16), we replace the DG-SEM problem by a lower order FV problem. This ansatz should not be confused with a p-multigrid method. It is motivated by the equivalence of the DG-SEM discretization to specific FV discretizations [19]. Instead of constructing the equivalent FV operator, we construct a first order FV replacement operator. This simplifies the construction of the Jacobian and allows us to use the available knowledge about fast multigrid methods for FV discretizations. The question arises how to transfer the degrees of freedom (DOFs) back and forth in these two discretizations inside each DG element. We present different transfer strategies later in this section and explain first the construction of FV agglomeration multigrid preconditioners.

In the multigrid method considered in this paper, the coarse grid problems are given by the FV discretization on the corresponding grid. The grid transfer is based on agglomeration: to restrict fine grid values, they are summed, weighted by the volumes of the respective cells and divided by the total volume. This is denoted by the restriction operator $\mathbf{R}^l_{l-1}$, where l indicates the grid level s.t. $l = 0$ is the coarsest level. The corresponding prolongation operator $\mathbf{P}^{l-1}_l$ is defined via injection, where the value in the coarse cell is taken for all corresponding fine cells.

A generic smoother is an iterative method for the solution of a linear system $\mathbf{A}_l\mathbf{x}_l = \mathbf{b}_l$ given by

$$\mathbf{x}^{k+1}_l = \mathbf{M}_{S,l}\mathbf{x}^k_l + \mathbf{N}^{-1}_{S,l}\mathbf{b}_l, \tag{17}$$

with an iteration matrix $\mathbf{M}_{S,l}$ and a nonsingular matrix $\mathbf{N}_{S,l}^{-1}$. As before, the index l specifies the multigrid level and S denotes that this iterative method represents a smoother. It is possible to construct an MG preconditioner based on a V- or a W-cycle, as well as several consecutive cycles. The number of pre- and postsmoothing steps is also flexible. A multigrid algorithm to solve the linear problem $\mathbf{A}\mathbf{x} = \mathbf{b}$ is presented in the following pseudo-code:

```
function MG($\mathbf{x}_l, \mathbf{b}_l, l$):
if($l = 0$)  solve  $\mathbf{A}_l\mathbf{x}_l = \mathbf{b}_l$
else:
    • for $pre = 1, \dots, N_{pre}$: $\mathbf{x}_l = \mathbf{M}_{S,l}\mathbf{x}_l + \mathbf{N}_{S,l}^{-1}\mathbf{b}_l$ (presmoothing)
  if($l > 0$)
    • $\mathbf{r}_{l-1} = \mathbf{R}_{l-1}^{l}(\mathbf{A}_l\mathbf{x}_l - \mathbf{b}_l)$ (restriction)
    • $\mathbf{v}_{l-1} = 0$
        – for $j = 1, \dots, \gamma$: $\mathbf{v}_{l-1} = MG(\mathbf{v}_{l-1}, \mathbf{r}_{l-1}, l-1)$
    • $\mathbf{s}_l = \mathbf{s}_l - \mathbf{P}_l^{l-1}\mathbf{v}_{l-1}$ (fine grid correction)
    • for $post = 1, \dots, N_{post}$: $\mathbf{s}_l = \mathbf{M}_{S,l}\mathbf{x}_l + \mathbf{N}_{S,l}^{-1}\mathbf{b}_l$ (postsmoothing)
```

This gives rise to an iterative method of the form

$$\mathbf{x}_l^{k+1} = \mathbf{M}_{MG}\mathbf{x}_l^{k} + \mathbf{N}_{MG}^{-1}\mathbf{b}_l. \tag{18}$$

The multigrid preconditioner is then given by $\mathbf{N}_{MG}^{-1}$.

3.1 Smoothers: Pseudo time stepping methods

As smoothers for the multigrid preconditioner we consider Runge-Kutta schemes. Adding a pseudo time derivative to an initial value problem $\mathbf{A}\mathbf{x} = \mathbf{b},\ \mathbf{x}(0) = \mathbf{x}^0$ yields

$$\frac{\partial \mathbf{x}}{\partial t^*} + \mathbf{A}\mathbf{x} - \mathbf{b} = \mathbf{0}. \tag{19}$$

Then any time-stepping method becomes a smoother. In this paper we consider low storage explicit s-stage Runge-Kutta schemes of the form

$$\begin{aligned} \mathbf{x}_j &= \mathbf{x}^n + \Delta t^* \alpha_j c(\mathbf{b} - \mathbf{A}\mathbf{x}^n),\ j = 1, \dots, s \\ \mathbf{x}^{n+1} &= \mathbf{x}^n + \Delta t^* c^*(\mathbf{b} - \mathbf{A}\mathbf{x}_s), \end{aligned} \tag{20}$$

with $\Delta t^* = \frac{c^*}{CFL_{adv} + CFL_{diff}}$, $CFL_{adv} =:= \frac{|\mathbf{b}|\Delta t*}{\Delta \mathbf{x}}$, $CFL_{diff} := \frac{2\varepsilon\Delta t*}{\Delta \mathbf{x}^2}$ and parameters α_j and c^*.

3.2 Transfer between DG and FV discretizations

In order to construct an FV replacement operator, the DG-SEM DOFs need to be reinterpreted as FV DOFs. In the following we discuss three different options to transfer the vector of unknowns between these two discretizations inside each DG element. The vector of unknowns in the DG-SEM discretization in one element is given by $\tilde{\mathbf{u}}$ and we define the corresponding vector in the FV discretization by $\hat{\mathbf{u}}$, where the number of volumes corresponds to the number of LGL nodes.

The first transfer option is referred to as *ad hoc assignment* of LGL nodes and FV cells in this paper and corresponds to the grid transfer presented in [13]. The idea is to assign the values of $\tilde{\mathbf{u}}$ to the values

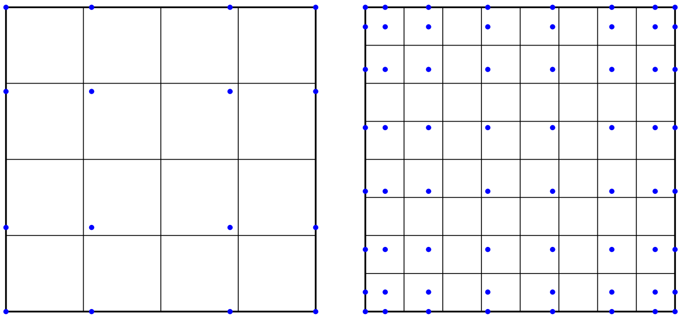

Figure 1: Two DG-SEM grid reference elements with LGL nodes (blue nodes), and an inlaid finite volume subgrid (black lines)

of $\hat{\mathbf{u}}$ based on the location of the DOF, i.e. each DG-SEM DOF is assigned to the closest FV DOF. A permutation matrix might be needed to take care of possible differences in the ordering of the DOFs. The distribution of 4 respective 8 LGL nodes and an overlaid equidistant FV grid is visualized in Figure 1. If every subgrid volume contains exactly one LGL node, a sufficient ad hoc assignment of DOFs between the two discretizations is provided, while for high order DG-SEM discretizations the FV subgrid may have volumes which do not contain any LGL node. Thus, some FV DOFs are assigned wrong values for their location. This could be improved by modifying the FV subgrid such that each volume contains exactly one LGL node.

Since the natural norm for DG methods is the L_2 norm, another idea for the space transfer is to consider an L_2 projection. To define it, either the basis functions of the FV space or the DG-SEM space can be used.

The basis functions in the FV space are already orthonormalized, and we define a projection matrix via $\mathbf{P}_{FV}\tilde{\mathbf{u}} = \hat{\mathbf{u}}$, defined as the inner L_2 product of the DG-SEM Lagrange basis and the FV basis functions. To transfer the DOFs from the FV grid to the DG grid, the reconstruction $\mathbf{P}_{FV}^{-1}$ can be used. This is not an L_2 projection, but gives good results in the numerical tests presented in the next section.

Another option is to define the L_2 projection matrix based on the DG-SEM space. To this end, the LGL basis has to be orthonormalized. Then the projection $\mathbf{P}_{DG}\hat{\mathbf{u}} = \tilde{\mathbf{u}}$ can be defined as $\mathbf{P}_{DG} := \mathbf{M}^{-1}\mathbf{P}_{FV}^{T}$ with the DG mass matrix $\mathbf{M}$ and the reverse transfer as $\mathbf{P}_{DG}^{-1}$.

4 NUMERICAL RESULTS

We test the efficiency of the presented preconditioner for the two-dimensional convection diffusion equation with $\mathbf{b} = \frac{1}{2}(\sqrt{3}, 1)$ and $\varepsilon = 1.e-4$. We show 40 iterations of the GMRES convergence rate using different FV based multigrid preconditioners. The number of multigrid levels is chosen such that the coarsest grid in the FV space corresponds to one DG cell. On the coarsest grid, the system is solved exactly using the sparse python solver `spsolve` which is based on an LU decomposition. For smoothing we use a 3-stage Runge-Kutta scheme with parameters $\alpha = [0.145, 0.395]$ and $c^* = 1.1$ [20]. We discretized the convection diffusion problem with a 4th and 8th order DG discretization and consider a $2^7 \times 2^7$ grid on the finest level. The time stepping width is set to be $\Delta t = 0.75$, which gives a hyperbolic CFL-like ratio of $\Delta t/\Delta x = 48$.

As a first numerical test we examine how accurate the replacement operator approximates the original problem and how efficient the different transfer operators are. The convergence results for the reference preconditioner based on the replacement first order FV operator can be seen in Figure 2, on the left side for a 4th order DG method, in the following referred to as DG4, and on the right side for a 8th order DG method referred to as DG8. In both cases, L_2 projection gives better results than the ad hoc assignment. Since the L_2 norm is used for DG discretizations, these are the results we expected. L_2 projection on the FV space gives slightly better results, with a difference in residual of approximately $1.e-1$ after 40 GMRES iterations both for DG4 and DG8. Increasing the order of the DG method gives results with a more similar convergence rate for all transfer options while the overall quality of the FV reference preconditioner decreases from between $1e-7$ and $1e-11$ to between $1e-3$ and $1e-5$. Since the reference operator is of first order, it had to be expected that the reference preconditioner works better for lower DG discretizations.

Next, we test the suggested multigrid preconditioners. A good multigrid method should approximate the reference operator as good as possible. We construct a 3-level preconditioner based on the FV replacement operator and test different combinations of pre- and postsmoothing for one V-cycle for DG4 and a 4-level preconditioner for DG8. In the legend of the plots, the first two numbers specify the pre- and postsmoothing steps on the finest grid and the last two numbers the pre- and postsmoothing steps on the coarser grids. On the coarsest grid we solve the system directly without any smoothing steps. As before, on the left side we show the results for the 4th and on the right side for the 8th order DG discretization.

In Figure 3 the convergence results can be seen using the ad hoc transfer between the DG and FV space. We notice that the suggested multigrid preconditioners give convergence rates which mimic the FV reference preconditioner to some extent and the preconditioner works better than an RK3 preconditioner. Nevertheless, the overall convergence improvement is not very good when increasing the DG order, for the reference preconditioner we decrease the relative residual by $1e-7$ in 40 iterations for a 4th order method compared to decreasing the relative residual by $1e-3$ in 40 iterations for a 8th order method. Moreover, the efficiency of the multigrid preconditioner is highly influenced by the number of smoothing steps on the finer grid levels.

In Figure 4 we consider interpolation by an L_2 projection on the FV grid. This transfer gives overall better results than the previous one, for the reference preconditioner we get a decrease the relative residual by $1e-11$ in 40 iterations for a 4th order method compared to decreasing the in relative residual by $1e-5$ for a 8th order method. With the suggested multigrid preconditioners we can decrease the relative residual between $1e-5$ and $1e-9$ for the 4th order DG method and between $1e-1$ and $1e-4$ for the 8th order DG method. Again it is of benefit to apply the smoother several times on the finest grid in order to get a convergence rate close to the one of the reference preconditioner. Moreover, postsmoothing on the finest grid improves the convergence rate. In comparison to the ad hoc transfer, the influence of the smoother on the multigrid preconditioner is more visible in this case and results in convergence rates with a difference of approximately one order of magnitude. The RK3 preconditioner does not perform well.

The results using an L_2 projection on the DG grid can be seen in Figure 5. The overall convergence improvement is about one order of magnitude worse than for the other L_2 projection both for the 4th and 8th order method. But we notice that the reference preconditioner is better approximated by the suggested multigrid preconditioners and the difference between the tested smoothing strategies in the multigrid preconditioners decreases. For the 4th order method, the relative residual can be decreased between $1e-7$ and $1e-9$ and for the 8th order method between $1e-3$ and $1e-3$. Again, presmoothing

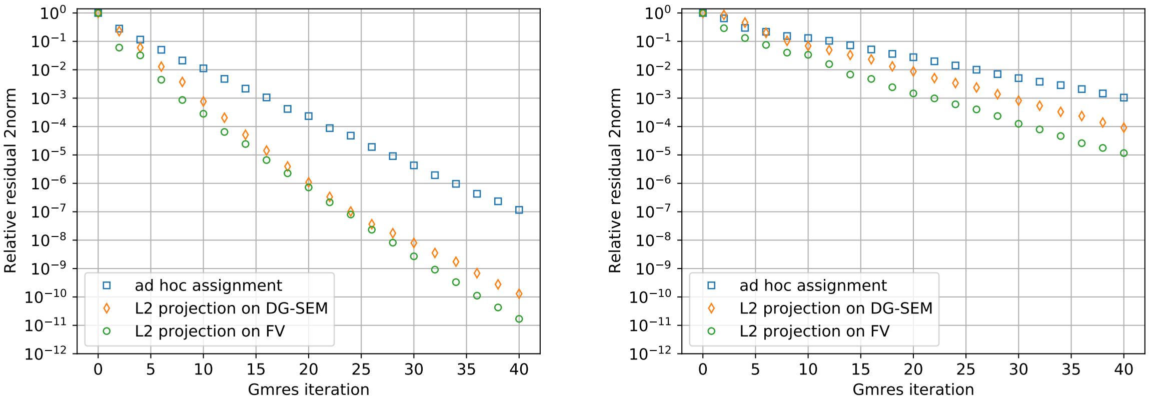

Figure 2: GMRES convergence rate for the reference FV preconditioner for different transfer strategies, 16384 DOFs for DG4 (left) and DG8 (right)

on the finest grid results in the best preconditioner. As before, the RK3 preconditioner is beaten by the multigrid preconditioners.

4.1 CPU time

Since the efficiency of the preconditioners is also influenced by their computational effort, we consider the corresponding CPU times. The results can be seen in Tables 1 and 2. As expected, applying the smoother on different levels has a direct impact on the CPU time. The multigrid based preconditioners are of very similar cost for different pre- and postsmoothing strategies, with more smoothing on the finest grid being more expensive. While the convergence rates are highly influenced by the choice of the transfer functions, this has not a huge impact on the measured CPU time. The same holds for increasing the order of the DG method. Setting the CPU times in relation to the convergence plots discussed before shows that the multigrid based preconditioner with L_2 projection on DG-SEM is most efficient.

Table 1: CPU time in seconds for 40 GMRES iterations, DG4 and 16384 DOFs

	ad hoc assignment	L2 on DG-SEM	L2 on FV
no prec.	6.39e-1	5.74e-1	6.69e-1
4× smoother	6.66e0	6.99e0	6.72e0
10,11	1.08e1	1.10e1	1.12e1
01, 11	1.13e1	1.11e1	1.09e1
22, 11	3.10e1	2.78e1	2.72e1
exact FV	4.90e0	4.82e0	5.06e0

5 CONCLUSIONS

In this paper we extended a multigrid preconditioning strategy for use in Jacobian-free Newton-Krylov methods for the solution of equation systems arising from implicit DG discretizations as presented in [13] to the two-dimensional case. The core idea is to make use of an auxiliary first order

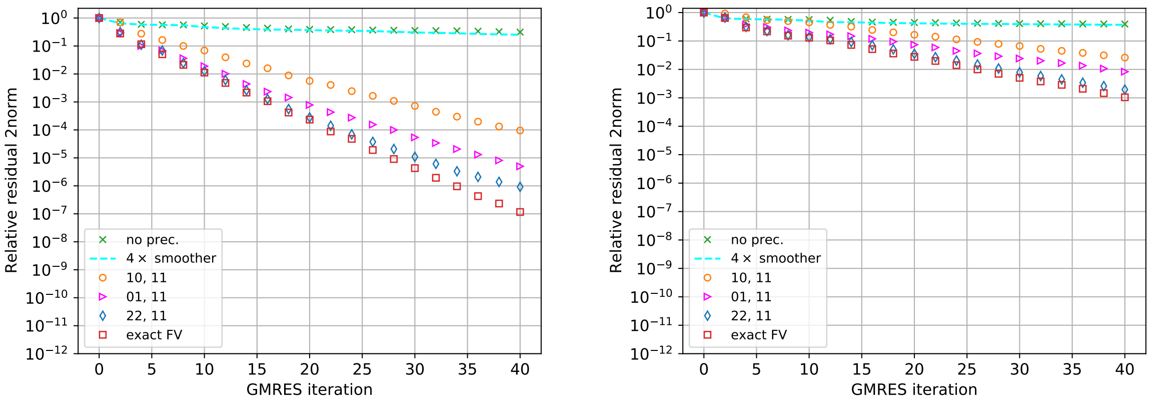

Figure 3: GMRES convergence rate for transfer via ad hoc assignment, 16384 DOFs for DG4 (left) and DG8 (right)

Table 2: CPU time in seconds for 40 GMRES iterations, DG8 and 16384 DOFs

	ad hoc assignment	L2 on DG-SEM	L2 on FV
no prec.	5.11e-1	5.47e-1	5.78e-1
4× smoother	6.22e0	6.82e0	7.17e0
10,11	1.48e1	1.52e1	1.53e1
01, 11	1.49e1	1.48e1	1.62e1
22, 11	3.27e1	3.17e1	3.32e1
exact FV	5.14e0	5.51e0	5.10e0

FV discretization that refines the original DG mesh, but can still be implemented algebraically. As smoother, we considered a 3-stage Runge-Kutta method. Numerical results for the two-dimensional convection-diffusion equation were presented. The convergence results of the proposed preconditioner are promising, being close to a quasi-exact preconditioner. Our results indicate that the performance of the preconditioner is highly influenced by the choice of grid transfer between DG and FV as well as on the order of the DG method. An L_2 projection improves the convergence rate compared to a simple ad hoc assignment. The projection can be precomputed for a given DG order, either based on the DG space or the FV space. For the presented problem, the L_2 projection on the FV grid is slightly more efficient than an L_2 projection on the DG-SEM grid. With both strategies we can decrease the GMRES residual from not even $1.e-1$ without preconditioner to approximately $1.e-5$-$1.e-11$ for a 4th order DG method and approximately $1.e-1$-$1.e-5$ for a 8th order DG method with the suggested multigrid preconditioners for the here presented L_2 grid transfer operators. Furthermore, our numerical experiments show the importance of the smoothing strategy in the multigrid method, both for the convergence rate and the CPU time. In order to improve the convergence results without increasing the CPU time drastically, more efficient and advanced smoothers can be used, for example a W3 smoother as presented in [13].

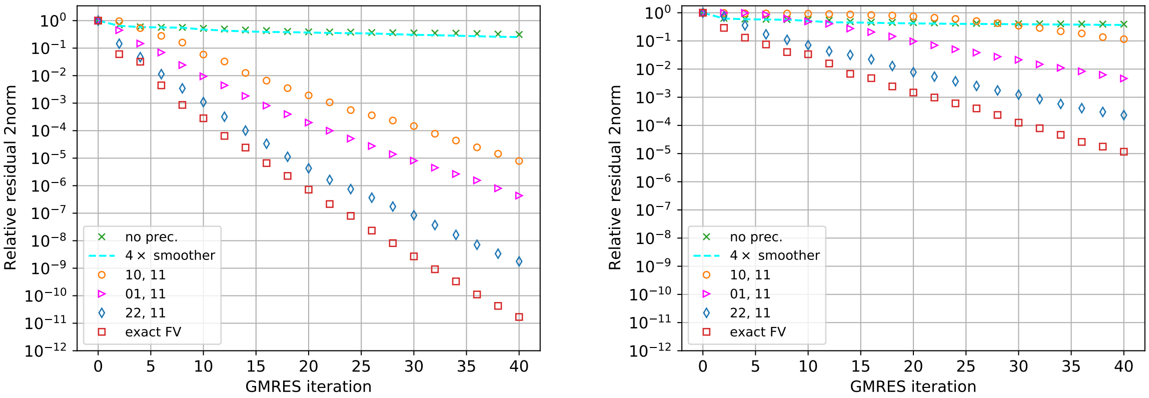

Figure 4: GMRES convergence rate for transfer via L2 projection on FV grid, 16384 DOFs for DG4 (left) and DG8 (right)

Funding

Philipp Birken and Lea Versbach gratefully acknowledge the support of the Swedish Research Council, grant number 2015-04133. Gregor Gassner has been supported by the European Research Council (ERC) under the EUs 8ths Framework Program Horizon 2020 with the research project Extreme, ERC grant agreement no. 714487.

REFERENCES

[1] D. A. Kopriva. *Implementing Spectral Methods for Partial Differential Equations*. Springer, 2009.

[2] H. T. Huynh. A flux reconstruction approach to high-order schemes including discontinuous Galerkin methods. In *18th AIAA Computational Fluid Dynamics Conference*, page 4079, 2007.

[3] F. Hindenlang, G. J. Gassner, C. Altmann, A. Beck, M. Staudenmaier, and C.-D. Munz. Explicit discontinuous Galerkin methods for unsteady problems. *Comp. & Fluids*, 61:86–93, 2012.

[4] P. Vincent, F. Witherden, B. Vermeire, J. S. Park, and A. Iyer. Towards green aviation with python at petascale. In *SC'16: Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, pages 1–11. IEEE, 2016.

[5] D. A. Kopriva, S. L. Woodruff, and M. Y. Hussaini. Computation of electromagnetic scattering with a non-conforming discontinuous spectral element method. *Int. J. Numer. Methods Eng.*, 53(1):105–122, 2002.

[6] G. J. Gassner. A skew-symmetric discontinuous Galerkin spectral element discretization and its relation to SBP-SAT finite difference methods. *SIAM J. Sci. Comput.*, 35(3):A1233–A1253, 2013.

[7] P. E. Vincent and A. Jameson. Facilitating the adoption of unstructured high-order methods amongst a wider community of fluid dynamicists. *Math. Model. Nat. Phenom.*, 6(3):97–140, 2011.

[8] D. A. Knoll and D. E. Keyes. Jacobian-Free Newton-Krylov Methods: A Survey of Approaches and Applications. *J. Comput. Phys.*, 193(2):357397, 2004.

[9] P. Birken and A. Jameson. On nonlinear preconditioners in Newton-Krylov methods for unsteady flows. *Int. J. Numer. Methods Fluids*, 62(5):565–573, 2010.

[10] P. Birken, G. J. Gassner, M. Haas, and C.-D. Munz. Preconditioning for modal discontinuous

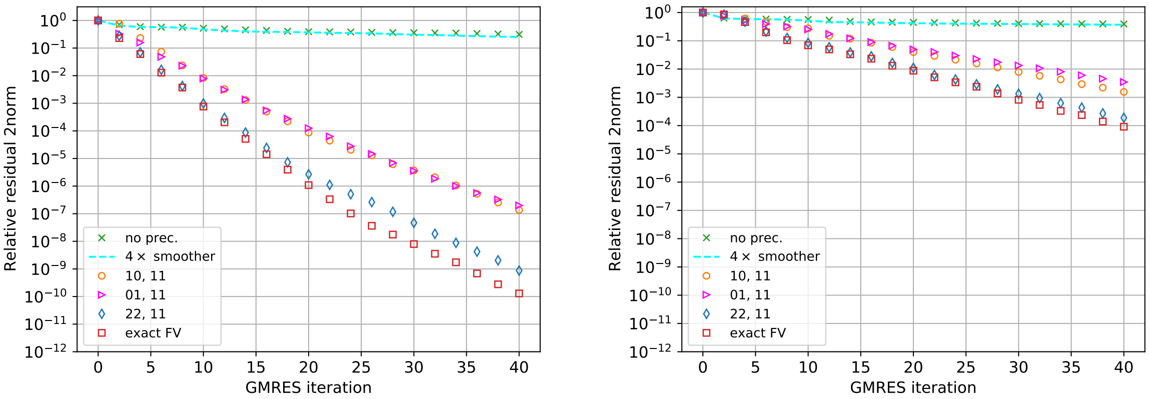

Figure 5: GMRES convergence rate for transfer via L2 projection on DG-SEM grid, 16384 DOF for DG4 (left) and DG8 (right)

Galerkin methods for unsteady 3D Navier-Stokes equations. *J. Comp. Phys.*, 240:20–35, 2013.

[11] P. Birken. *Numerical methods for the unsteady compressible Navier-Stokes equations, Habilitation Thesis*. Habilitation thesis, Universität Kassel, 2012.

[12] L. M. Versbach, P. Birken, and G. J. Gassner. Finite volume based multigrid preconditioners for discontinuous Galerkin methods. *PAMM*, 18(1):e201800203, 2018.

[13] P. Birken, G. J. Gassner, and L. M. Versbach. Subcell finite volume multigrid preconditioning for high-order discontinuous Galerkin methods. *Int. J. Comput. Fluid Dyn.*, pages 1–9, 2019.

[14] K. J. Fidkowski, J. Oliver, T. A .and Lu, and D. L. Darmofal. p-Multigrid solution of high-order discontinuous Galerkin discretizations of the compressible Navier–Stokes equations. *J. Comput. Phys.*, 207(1):92–113, 2005.

[15] Y. Allaneau, L. Y. Li, and A. Jameson. Convergence Acceleration of High Order Numerical Simulations using a Hybrid Spectral Difference-Finite Volume Multigrid Method. In *7th International Conference on Computational Fluid Dynamics (ICCFD7)*, pages 9–13, 2012.

[16] P. Bastian, M. Blatt, M. Dedner, N.-A. Dreier, R. Engwer, Ch. Fritze, C. Gräser, Ch. Grüninger, D. Kempf, R. Klöfkorn, M. Ohlberger, and O. Sander. The Dune framework: Basic concepts and recent developments. *CAMWA*, 2020.

[17] G. Karniadakis and S. Sherwin. *Spectral/hp element methods for computational fluid dynamics*. Oxford University Press, 2013.

[18] S. Brdar, A. Dedner, and R. Klöfkorn. Compact and stable Discontinuous Galerkin methods for convection-diffusion problems. *SIAM J. Sci. Comput.*, 34(1):A263–A282, 2012.

[19] T. C. Fisher and M. H. Carpenter. High-order entropy stable finite difference schemes for nonlinear conservation laws: Finite domains. *J. Comput. Phys.*, 252:518–557, 2013.

[20] P. Birken. Optimizing Runge-Kutta smoothers for unsteady flow problems. *Electron. Trans. Numer. Anal.*, 39(1):298–312, 2012.