ECCOMAS Congress 2016
VII European Congress on Computational Methods in Applied Sciences and Engineering
M. Papadrakakis, V. Papadopoulos, G. Stefanou, V. Plevris (eds.)
Crete Island, Greece, 5–10 June 2016

A STUDY OF MULTIGRID SMOOTHERS USED IN COMPRESSIBLE CFD BASED ON THE CONVECTION DIFFUSION EQUATION

Philipp Birken[1], **Jonathan Bull**[2], **and Antony Jameson**[3]

[1]Lund University, Centre for the Mathematical Sciences, Numerical Analysis
Box 118, 22100 Lund
e-mail: philipp.birken@na.lu.se

[2] Uppsala University, Division of Scientific Computing, Dept. of Information Technology
Box 337, 75105 Uppsala
e-mail: jonathan.bull@it.uu.se

[3] Stanford University, Department of Aeronautics & Astronautics
Stanford, CA 94305
e-mail: jameson@baboon.stanford.edu

Keywords: Unsteady flows, Multigrid

Abstract. *We look at multigrid methods for unsteady viscous compressible flows. We specifically target smoothers that can be used in parallel and without computation of a Jacobian, which are particlarly attractive candidates in the context of Discontinuous Galerkin discretizations. In CFD, a plethora of nonlinear smoothers have been suggested which are hard to analyze. Our methodology is to use a linear model problem, here the convection diffusion equation, to be able to classify and compare smoothers better. Specifically, we consider explicit and implicit pseudo time iterations, GMRES as a smoother, SGS and implicit line smoothers. We relate GMRES to explicit Runge-Kutta smoothers, identify implicit line smoothers as Block Jacobi and analyze the potential of implicit pseudo time iterations. Finally, we discuss the relation between methods for steady and unsteady flows. Numerical results show that GMRES is a very attractive smoother in this context.*

1 INTRODUCTION

After decades of research, numerical methods for solving steady fluid flows have reached an appreciable level of maturity. It has been demonstrated that steady Euler flows can be solved with 'textbook' efficiency in three to five multigrid iterations [3]. The added stiffness of the steady RANS equations means that convergence of the multigrid method is somewhat slower - between 50 and 100 iterations [11] - but still within acceptable bounds for CFD applications.

For unsteady problems, the expectation would be to significantly reduce this number, but this is not the case for the available methods [2]. When looking at high order methods, which will be necessary to do LES simulations in an industrial setting, there is the situation that a fast multigrid method is quite simply missing. On reason for this is the lack of guiding theory for nonelliptic equations, which makes designing a multigrid method difficult.

Multigrid consists of two components, the smoother and the coarse grid correction. For finite volume methods, the coarse grid correction is based on agglomeration of neighboring cells. As smoothers for compressible flows, a plethora of both linear and nonlinear methods has been suggested. This ranges from LU-SGS [3], explicit Runge-Kutta methods [13], additive Runge-Kutta methods [5], implicit line smoothers [12, 8], and more recently point implicit and SDIRK smoothers [7, 9], GMRES [4], GMRES within a Rosenbrock pseudo time iteration [10] and many others. In particular, implicit smoothers, that are variants of implicit pseudo time iteration smoothers, allow for a huge variety of methods.

When thinking of LES simulations with discontinuous Galerkin methods, we have to keep in mind that the Jacobians will consist of blocks that have a size of a few hundred, instead of just five as it would be with finite volume methods. Our strategy to obtain a parallel scaling, fast and low memory multigrid method is therefore to not use a Jacobian, but only function evaluations, meaning evaluations of the spatial discretization. This can happen in one of two ways: There is a nonlinear version of a linear iterative method that avoids use of the Jacobian. This is for example the case for pseudo time iterations where a matrix vector product corresponds to a function evaluation. The other option is to replace matrix vector products by a Jacobian free finite difference.

Previous work by one of the authors [1] demonstrated that the smoothers used in the multigrid iteration could be optimized for fastest convergence of the unsteady linear advection equation. Specifically, it was suggested to numerically minimize the spectral radius of the multigrid iteration matrix over the pseudo timestep and coefficients of explicit Runge-Kutta (RK) schemes. Significant improvements were demonstrated in both cases compared to non-optimized smoothers.

In this paper we aim at getting a better understanding of the differences between these methods for both the steady and the unsteady case. To this end, we classify these methods, The findings will be backed up by numerical results based on the convection diffusion equation as a model problem.

2 GOVERNING EQUATIONS AND DISCRETIZATION

We consider the linear advection diffusion equation

$$u_t + au_x - bu_{xx} = 0. \tag{1}$$

with $a, b > 0$ on the interval $x \in [0, 2]$ with periodic boundary conditions.

An equidistant FV discretization for (1) with mesh width Δx leads to the evolution equation

for the cell average u_i in one cell i:

$$u_{i_t} + \frac{a}{\Delta x}(u_i - u_{i-1}) + \frac{b}{\Delta x^2}(-u_{i+1} + 2u_i - u_{i-1}) = 0.$$

Using the vector $\mathbf{u} = (u_1, ..., u_m)^T$ and the matrices

$$\mathbf{B} = \begin{pmatrix} 1 & & & & -1 \\ -1 & 1 & & & \\ & -1 & 1 & & \\ & & \ddots & \ddots & \\ & & & -1 & 1 \end{pmatrix}$$

and

$$\mathbf{C} = \begin{pmatrix} 2 & -1 & & & -1 \\ -1 & 2 & -1 & & \\ & -1 & 2 & \ddots & \\ & & \ddots & \ddots & -1 \\ -1 & & & -1 & 2 \end{pmatrix}$$

we obtain the system of ODEs

$$\mathbf{u}_t + \frac{a}{\Delta x}\mathbf{B}\mathbf{u}(t) + \frac{b}{\Delta x^2}\mathbf{C}\mathbf{u}(t) = \mathbf{0}. \tag{2}$$

In 2D, we have the following equation for the unknown function $u(x, y, t)$:

$$u_t + a \cdot \nabla u - b\Delta u = 0, \tag{3}$$

where $b > 0$ and

$$a = \tilde{a} \begin{pmatrix} \sin\gamma \\ \cos\gamma \end{pmatrix}$$

with $\tilde{\beta} \in \mathbf{R}$ being another parameter and γ the angle of the direction of forced convection. Regarding boundary conditions, we use periodic ones.

An equidistant FV discretization for (3) with mesh width $\Delta x = \Delta y$ leads to the evolution equation for the cell average $u_{i,j}$ in one cell (i, j):

$$u_{i,j_t} + \frac{\tilde{a}}{\Delta x}((c+s)u_{i,j} - su_{i-1,j} - cu_{i,j-1})$$
$$+\frac{b}{\Delta x^2}(-u_{i+1,j} - u_{i,j+1} + 4u_{i,j} - u_{i-1,j} - u_{i,j-1}) = 0.$$

Using the vector $\mathbf{u} = (u_1, ..., u_m)^T$ and the matrices

$$\mathbf{B} = \begin{pmatrix} \bar{\mathbf{B}} & -s\mathbf{I} & & \\ & \bar{\mathbf{B}} & \ddots & \\ & & \ddots & -s\mathbf{I} \\ -s\mathbf{I} & & & \bar{\mathbf{B}} \end{pmatrix},$$

where

$$\bar{\mathbf{B}} = \begin{pmatrix} c+s & & & & -c \\ -c & c+s & & & \\ & -c & c+s & & \\ & & \ddots & \ddots & \\ & & & -c & c+s \end{pmatrix}$$

and

$$\mathbf{C} = \begin{pmatrix} \bar{\mathbf{C}} & -\mathbf{I} & & -\mathbf{I} \\ -\mathbf{I} & \bar{\mathbf{C}} & -\mathbf{I} & \\ & \ddots & \ddots & \ddots \\ -\mathbf{I} & & -\mathbf{I} & \bar{\mathbf{C}} \end{pmatrix},$$

where

$$\bar{\mathbf{C}} = \begin{pmatrix} 4 & -1 & & & -1 \\ -1 & 4 & -1 & & \\ & -1 & 4 & \ddots & \\ & & \ddots & \ddots & -1 \\ -1 & & & -1 & 4 \end{pmatrix}.$$

Finally, we obtain the system of ODEs

$$\mathbf{u}_t + \frac{\tilde{a}}{\Delta x}\mathbf{B}\mathbf{u}(t) + \frac{b}{\Delta x^2}\mathbf{C}\mathbf{u}(t) = \mathbf{0}. \tag{4}$$

We discretize this in time using implicit Euler with time step size Δt, which is also a building block for the more general diagonally implicit Runge-Kutta (DIRK) methods. Thus, in each time step, a linear system has to be solved. Using the notation $\mathbf{u}^n \approx \mathbf{u}(t_n)$, this can be written as

$$\begin{aligned} \mathbf{u}^{n+1} - \mathbf{u}^n + \frac{\tilde{a}\Delta t}{\Delta x}\mathbf{B}\mathbf{u}^{n+1} + \frac{b\Delta t}{\Delta x^2}\mathbf{C}\mathbf{u}^{n+1} &= \mathbf{0} \\ \Leftrightarrow \mathbf{u}^n - \mathbf{A}\mathbf{u}^{n+1} &= 0 \end{aligned} \tag{5}$$

where

$$\mathbf{A} = \left(\mathbf{I} + \frac{\nu}{\Delta x}\mathbf{B} + \frac{\mu}{\Delta x^2}\mathbf{C}\right) \tag{6}$$

with $\nu = \tilde{a}\Delta t$ and $\mu = b\Delta t$. Here, $CFL := \tilde{a}\Delta t/\Delta x$ corresponds to the CFL number in the implicit Euler method. If we consider nonperiodic boundary conditions, the entry in the upper right corner of $\mathbf{B}$ becomes zero. Otherwise, additional terms appear on the right hand side, but this does not affect multigrid convergence.

In a steady state, the time dependence vanishes and we obtain the equation system

$$\frac{a}{\Delta x}\mathbf{B}\mathbf{u} = \mathbf{0}. \tag{7}$$

3 LINEAR ITERATIVE METHODS FOR LINEAR EQUATION SYSTEMS

A very important type of iterative method for the linear equation system

$$\mathbf{A}\mathbf{x} = \mathbf{b}, \quad \mathbf{A} \in \mathbf{R}^{m\times m}, \mathbf{x}, \mathbf{b} \in \mathbf{R}^m \tag{8}$$

are linear iterative methods which can be written as

$$\mathbf{x}^{k+1} = \mathbf{M}\mathbf{x}^k + \mathbf{N}^{-1}\mathbf{b},\ \mathbf{M}, \mathbf{N} \in \mathbf{R}^{m\times m}.$$

It can be shown that this converges if and only of $\rho(\mathbf{M}) < 1$, which is why $\mathbf{M}$ is called the iteration matrix. The second matrix determines the limit in case of convergence. To get convergence to the solution of the linear equation system 8, it is required that $\mathbf{M} = \mathbf{I} - \mathbf{N}^{-1}\mathbf{A}$.

4 MULTIGRID METHOD

As a solver, an agglomeration multigrid method is used, which corresponds best to finite volume discretizations. Thus, in the one dimensional case, the restriction and prolongation correspond to joining and dividing neighboring cells and are given by

$$\mathbf{R} = \frac{1}{2}\begin{pmatrix} 1 & 1 & & & & \\ & & 1 & 1 & & \\ & & & \ddots & \ddots & \\ & & & & 1 & 1 \end{pmatrix} \text{ and } \mathbf{P} = 2\mathbf{R}^T = \begin{pmatrix} 1 & & & & \\ 1 & & & & \\ & 1 & & & \\ & 1 & & & \\ & & \ddots & & \\ & & & 1 & \\ & & & 1 & \end{pmatrix}.$$

Hereby, it is assumed that we have an even number of cells.

The coarse grid matrix is obtained by discretizing the problem on that grid. On the coarsest level, the smoother is applied instead of the usual direct solver, since this better corresponds to the Full Approximation Scheme used for the nonlinear equations. We use a V-cycle and presmoothing only. Thus we obtain the scheme:

Function MG$(\mathbf{x}_l, \mathbf{b}_l, l)$

- $\mathbf{x}_l = \mathbf{S}_l^{\nu_1}(\mathbf{x}_l, \mathbf{b}_l)$ (Presmoothing)
- if $(l > 0)$
  - $\mathbf{r}_{l-1} = \mathbf{R}_{l-1,l}(\mathbf{b}_l - \mathbf{A}_l\mathbf{x}_l)$ (Restriction)
  - $\mathbf{v}_{l-1} = 0$
  - Call $MG(\mathbf{v}_{l-1}, \mathbf{r}_{l-1}, l-1)$ (Computation of the coarse grid correction)
  - $\mathbf{x}_l = \mathbf{x}_l + \mathbf{P}_{l,l-1}\mathbf{v}_{l-1}$ (Correction via Prolongation)
- end if

Thus, the iteration matrix of the corresponding two grid scheme is given by

$$\mathbf{M} = (\mathbf{I} - \mathbf{P}_{l,l-1}(\mathbf{N}_S^{l-1})^{-1}\mathbf{R}_{l-1,l}\mathbf{A}_l)\mathbf{M}_S^l \tag{9}$$

with $\mathbf{M}_S^l$ and $(\mathbf{N}_S^l)^{-1}$ being the matrices defining the smoother. In a standard two grid method, the matrix $\mathbf{N}_S^{-1}$ would be $\mathbf{A}_{l-1}^{-1}$ instead.

	α_1	α_2	α_3	c
ERK2	0.28			1.1
ERK4	0.04	0.12	0.32	2.0

Table 1: Coefficients for 2- and 4-stage ERK smoothers.

4.1 Smoothers

We consider a smoother to be good if the spectral radius of the iteration matrix is small. As a design criterion, this results in good smoothers, but it is expensive to use [1]. The main alternative is the smoothing factor, which is the slowest rate at which the error in one of the high frequency components is reduced by the smoother. To this end, the eigenvectors of $\mathbf{A}$ are determined and split into low and high frequency components.

4.1.1 Pseudo time iterations

The first class of smoothers we consider are time integration schemes. These approximate the solution of an initial value problem

$$\mathbf{u}_t = \mathbf{f}(\mathbf{u}), \quad \mathbf{u}_n = \mathbf{u}(t_n).$$

To apply them as iterative solvers for a linear equation system, the residual of that is used as the right hand side $\mathbf{f}(\mathbf{u})$. The differential equation resulting from (5) is given in a pseudo or dual time t^*:

$$\mathbf{u}_{t^*} = \mathbf{u}^n - \mathbf{A}\mathbf{u}(t^*), \quad \mathbf{u}(t_0^*) = \mathbf{u}^n. \tag{10}$$

One step of a pseudo time iteration thus consists of performing one step of the RK scheme for the solution of equation (10). The pseudo time step Δt^* is a parameter of this class of smoothers, as are their coefficients. However, instead of Δt^*, we will use the CFL number in pseudo time $c = a\Delta t^*/\Delta x$ as a parameter. This implies that on coarser grids, larger time steps will be chosen, leading to faster multigrid methods.

The simplest smoothers of this type are low storage explicit Runge-Kutta methods, which can be implemented using only three vectors. These are described by

$$\begin{aligned}\mathbf{u}_0 &= \mathbf{u}_n\\ \mathbf{u}_j &= \mathbf{u}_n + \alpha_j \Delta t^* \mathbf{f}(\mathbf{u}_{j-1}),\ j = 1, ..., s-1\\ \mathbf{u}_{n+1} &= \mathbf{u}_n + \Delta t^* \mathbf{f}(\mathbf{u}_{s-1}),\end{aligned}$$

For $s = 1$, we obtain he explicit Euler method, which is nothing but a Richardson smoother:

$$\mathbf{u}^{k+1} = \mathbf{u}^k + \Delta t^*(\mathbf{u}^n - \mathbf{A}\mathbf{u}^k) = \mathbf{u}^k + \Delta t^* \mathbf{r}^k. \tag{11}$$

In the linear case, a general Runge-Kutta method can be represented by its stability function $R(z)$, which for explicit Runge-Kutta methods is a polynomial $P_s(z)$ of degree s. Thus, the iteration matrix is for our case given by

$$\mathbf{M} = P_s(-\Delta t \mathbf{A}).$$

The specific schemes used here are a 2-stage and a 4-stage scheme taken from [1], where the coefficients for the case of a large outer time step are taken. These can be seen in Table 1.

	i	1	2	3	4	5	c
ARK4	α_i	1/3	4/15	5/9	1	-	0.5
	β_i	1	1/2	0	0	-	
ARK 5	α_i	1/4	1/6	3/8	1/2	1	0.2
	β_i	1	0	0.56	0	0.44	

Table 2: Coefficients of additive Runge-Kutta smoothers, 4-stage and 5-stage method.

In [6], Jameson, Schmidt and Turkel introduced additive Runge-Kutta methods, where different coefficients are used for the convective and the diffusive parts. This is done to reduce the number of evaluations of the expensive diffusive terms, but also to increase the degrees of freedom in choosing a good smoother. Given a splitting in the convective and diffusive part

$$\mathbf{f}(\mathbf{u}) = \mathbf{f}^c(\mathbf{u}) + \mathbf{f}^v(\mathbf{u}),$$

Jameson suggests to implement these schemes in the following equivalent form

$$\begin{aligned} \mathbf{u}_l^{(0)} &= \mathbf{u}_l \\ \mathbf{u}_l^{(j)} &= \mathbf{u}_l - \alpha_j \Delta t_l (\mathbf{f}^{c,(j-1)} + \mathbf{f}^{v,(j-1)}), \quad j = 1, ..., s \\ \mathbf{u}_l^{n+1} &= \mathbf{u}_l^{(s)}, \end{aligned}$$

where

$$\begin{aligned} \mathbf{f}^{c,(j)} &= \mathbf{f}^c(\mathbf{u}_l^{(j)}), \quad j = 0, ..., s-1 \\ \mathbf{f}^{v,(0)} &= \mathbf{f}^v(\mathbf{u}_l^{(0)}), \\ \mathbf{f}^{v,(j)} &= \beta_j \mathbf{f}^v(\mathbf{u}_l^{(j)}) + (1-\beta_j)\mathbf{f}^{v,(j-1)}, \quad j = 1, ..., s-1. \end{aligned}$$

For our model problems, we have $\mathbf{f}^c(\mathbf{u}) = \mathbf{b} - \mathbf{B}\mathbf{u}$ and $\mathbf{f}^v(\mathbf{u}) = -\mathbf{C}\mathbf{u}$. The iteration matrix now involves both powers of the single matrices, as well as products, making it very difficult to analyze. The schemes we consider were designed in [5] and the coefficients are given in Table 2.

The smoothing factor is the maximum of the stability function over the high frequency eigenvalues:

$$\max_{\lambda_{HF}} |R(\lambda)|.$$

Thus we look for methods with an optimal smoothing factor, found by

$$\min_{\alpha, \Delta t^*} \max_{\lambda_{HF}} |R(\lambda)|.$$

For an ERK scheme, this gives

$$\min_{\alpha, \Delta t^*} \max_{\lambda_{HF}} |P_s(\lambda)|. \tag{12}$$

The vector α contains the coefficients of the time integration method. Here, we will, instead of the pseudo time step Δt^*, use the CFL number in pseudo time, c^*, giving

$$\min_{\alpha, c^*} \max_{\lambda_{HF}} |R(\lambda)|. \tag{13}$$

In case of an implicit RK method, the stability function is rational, e.g. $R(z) = 1/(1+z)$ for the implicit Euler method. However, this implies that the application of the smoother consists of solving the original linear equation system, defying the purpose of an iterative method. Therefore, the arising system is solved using another iterative method, resulting in new families of smoothers. We will consider the particular case of the implicit Euler method, combined with GMRES.

4.2 GMRES

Given the matrix $\mathbf{A}$ and a residual vector $\mathbf{r}_0 = \mathbf{b} - \mathbf{A}\mathbf{x}^0$, the space

$$\mathcal{K}_l = \text{span}\{\mathbf{r}_0, \mathbf{A}\mathbf{r}_0, ..., \mathbf{A}^{l-1}\mathbf{r}_0\}$$

is called the l-th Krylov subspace. Using this, the k-th iteration of GMRES is the solution of the problem

$$\min_{x \in x^{(0)} + \mathcal{K}_k} \|\mathbf{b} - \mathbf{A}\mathbf{x}\|_2, \tag{14}$$

If $x \in x^{(0)} + \mathcal{K}_k$, then

$$\mathbf{x} = \mathbf{x}^{(0)} + \sum_{j=0}^{k-1} \gamma_j \mathbf{A}^j \mathbf{r}_0 = \mathbf{x}^{(0)} + \sum_{j=0}^{k-1} \gamma_j \mathbf{A}^j (\mathbf{b} - \mathbf{A}\mathbf{x}^0).$$

Thus, the iteration can be written as

$$\mathbf{x}^k = \sum_{j=0}^{k-1} \bar{\gamma}_j \mathbf{A}^j \mathbf{b} + \sum_{j=0}^{k} \bar{\gamma}_j \mathbf{A}^j \mathbf{x}^0$$

for some coefficients $\bar{\gamma}_j$ and the iteration matrix is

$$\mathbf{M}_k^{GMRES} = \sum_{j=0}^{k} \bar{\gamma}_j \mathbf{A}^j = p_k(\mathbf{A}).$$

Thus, the iteration for a GMRES method with fixed k steps is represented by a polynomial of degree k in $\mathbf{A}$ or otherwise put, a rational function is approximated by a polynomial. The difference to an explicit RK iteration is that the coefficients are not fixed a priori, but depend on the choice of Krylov subspace, in this case on the residual $\mathbf{r}_0$. This means that the smoother will behave differently for different initial guesses. This also means that it is difficult to analyze the smoothing factor. At least, from the definition of GMRES, we see that

$$\|\mathbf{r}_k\|_2 = \min_{p \in \Pi_k, p(0)=1} \|p(\mathbf{A})\mathbf{r}_0\|_2 \leq \|\bar{p}(\mathbf{A})\mathbf{r}_0\|_2,$$

which is similar to (12). The first difference is that here, the optimization takes into account all eigenvalues, whereas the previous one looks only at the high frequency ones. Then, the optimization here is dynamic in the sense that it is done automatically by GMRES for each linear system seperate, which makes this approach potentially more robust. For the optimized ERK schemes, it could happen that when operating in off design conditions, the performance significantly deteriorates.

When considering the use of GMRES within an implicit Runge-Kutta smoother, the following property is important. For $\mathbf{A} = \mathbf{I} + \alpha\mathbf{B}$,

$$\text{span}\{\mathbf{r}_0, \mathbf{A}\mathbf{r}_0, ..., \mathbf{A}^{l-1}\mathbf{r}_0\} = \text{span}\{\mathbf{r}_0, \mathbf{B}\mathbf{r}_0, ..., \mathbf{B}^{l-1}\mathbf{r}_0\}$$

and thus, for the same vector $\mathbf{r}_0$ the Krylov subspaces are identical. Now we look at the following two approaches:

1. Use k-step GMRES as smoother

2. Use a smoother which consists of one implicit Euler step, where the linear system is solved using k-step GMRES

In the first case, we look at the system

$$(\mathbf{I} + \Delta t\mathbf{B})\mathbf{u} = \mathbf{u}^n$$

with residual

$$\mathbf{r} = \mathbf{u}^n - (\mathbf{I} + \Delta t\mathbf{B})\mathbf{u}. \tag{15}$$

In the second case, the system is

$$[\mathbf{I} + \Delta t^*(\mathbf{I} + \Delta t\mathbf{B})]\mathbf{u} = \mathbf{u}^k + \Delta t^*\mathbf{u}^n$$

with residual

$$\mathbf{r}^* = \mathbf{u}^k + \Delta t^*\mathbf{u}^n - [\mathbf{I} + \Delta t^*(\mathbf{I} + \Delta t\mathbf{B})]\mathbf{u} = \mathbf{u}^k - \mathbf{u} + \Delta t^*\mathbf{r}. \tag{16}$$

By the property named above, the Krylov subspaces generated will be $\mathcal{K}_k(\mathbf{B}, \mathbf{r})$ and $\mathcal{K}_k(\mathbf{B}, \mathbf{u}^k - \mathbf{u} + \Delta t^*\mathbf{r})$, respectively. On the fine grid, the initial guess for the smoother is $\mathbf{u}^k$ and thus $\mathbf{r}_0^* = \mathbf{u}^k - \mathbf{u} + \Delta t^*\mathbf{r} = \Delta t^*\mathbf{r}_0$ and therefore, the two spaces are identical. However, the residuals (15) and (16) minimized by GMRES are different and therefore, unless the solution is in that space, a different result will be obtained. However, for a large value of Δt^*, this difference will vanish.

On the coarse grid, we have a zero initial guess for the equation $(\mathbf{I}+\Delta t\mathbf{B})\mathbf{e} = \mathbf{r}$ and therefore in the first case,

$$\mathbf{r}_0 = \mathbf{r} - (\mathbf{I} + \Delta t\mathbf{B})\mathbf{0} = \mathbf{r}.$$

For the second variant we have the equation

$$[\mathbf{I} + \Delta t^*(\mathbf{I} + \Delta t\mathbf{B})]\mathbf{e} = \mathbf{u}^k + \Delta t^*\mathbf{u}^n - [\mathbf{I} + \Delta t^*(\mathbf{I} + \Delta t\mathbf{B})]\mathbf{u}^{k+1}$$

and thus

$$\hat{\mathbf{r}}_0 = \mathbf{u}^k - \mathbf{u}^{k+1} + \Delta t^*\mathbf{r}.$$

Thus, on the coarse grids, the smoothers no longer operate in the same Krylov subspace. Again, for a large value of Δt^*, this difference will vanish. Thus, we expect the two smoothers to behave the same way for a sufficiently large pseudo time step.

4.3 Symmetric Gauss-Seidel

Given a splitting of the matrix $\mathbf{A}$ into its strictly lower, diagonal and strictly upper part, $\mathbf{A} = \mathbf{L} + \mathbf{D} + \mathbf{U}$, the symmetric Gauss-Seidel (SGS) iteration is given by

$$\mathbf{u}^{k+1} = (\mathbf{I} - (\mathbf{D} + \mathbf{U})^{-1}\mathbf{D}(\mathbf{L} + \mathbf{D})^{-1}\mathbf{A})\mathbf{u}^k + (\mathbf{D} + \mathbf{U})^{-1}\mathbf{D}(\mathbf{L} + \mathbf{D})^{-1}\mathbf{b},$$

thus the iteration matrix is

$$\mathbf{M}^{SGS} = \mathbf{I} - (\mathbf{D} + \mathbf{U})^{-1}\mathbf{D}(\mathbf{L} + \mathbf{D})^{-1}\mathbf{A}.$$

This method is parameter free and thus its smoothing factor depends on the problem only. SGS is implemented by solving two equation systems with triangular system matrices.

s-ARK	s-ERK	GMRES-k	SGS	ILS
$s+1$	s+1	$k+2$	3.5	5

Table 3: Computational effort of multigrid method in terms of function evaluations, respectively MatVecs.

4.4 Implicit line smoothers

For compressible viscous flows, implicit point smoothers and implicit line smoothers (ILS) are sometimes used. Both are in effect variants of block Jacobi smoothers, meaning that the following iteration is used

$$\mathbf{u}^{k+1} = (\mathbf{I} - \mathbf{D}^{-1}\mathbf{A})\mathbf{u}^k + \mathbf{D}^{-1}\mathbf{b},$$

where $\mathbf{D}$ is a certain block diagonal matrix. For the point implicit smoother, the small blocks of the size of the number of equations in the PDE system (four for 2D Navier-Stokes or Euler, 5 in 3D). Since the convection diffusion equation is scalar, this reduces to the normal Jacobi smoother, which is not a good method. Therefore, we only consider implicit line smoothers here.

These are used typically in boundary layers with lines going a finite length in a direction normal to the boundary. Then the block is defined from the unknowns in that line and has the size as the number of equations corresponding to these unknowns. Here, we will use in the 2D case lines going in x-direction through the the whole domain, meaning that there is exactly one line for each discrete y value and their length is the number of unknowns in x-direction. We denote this smoother as ILS.

4.5 Computational effort

The convergence rate of a multigrid method with a specific smoother is important, but it needs to be evaluated with regards to the computational effort needed. In the nonlinear case, the cost should be measured by the number of evaluations of the right hand side, involving computation of numerical fluxes and so on. In the linear case, this corresponds to matrix vector products (MatVecs). For an s-step ERK or a k-step GMRES smoother, the number of these is easy to determine, namely s and $k+1$. For an ARK method, it should be noted that while we actually increase computational effort in the linear case, this is not so in the nonlinear case, where the computation of viscous and inviscid fluxes is essentially independent. Thus, we consider the ARK smoother to have a computational effort proportional to s MatVecs as well.

With regards to SGS, there is one MatVec, one multiplication with a diagonal and two solves with tridiagonal equation systems. The latter take together slightly more than on MatVec, which is why we use 2.5 MatVecs as a measure for SGS. As for ILS, this requires solving a number of equation systems of smaller dimension. This is hard to predict, which is why we compared the CPU time for the use of that to one MatVec, arriving at a factor of four. Added to cost of a smoother in the present context is the cost of the MatVec needed for the computation of the residual on the fine grid.

4.6 Steady versus unsteady

We now consider the case of a parameter dependent smoother with parameter ω for the linear equation system $\mathbf{Ax} = \mathbf{b}$. This could be a damping parameter or a pseudo time step size, respectively CFL number in pseudotime. The iteration matrix is then of the form $\mathbf{M}^s(\omega\mathbf{A})$. Assume that we have determined an optimal coefficient ω^* for a steady state problem. For an

c	ImpE	GMRES-2
1	0.4903	0.3772
10	0.3798	0.3772
100	0.3774	0.3772
1000	0.3773	0.3772

Table 4: Measured convergence rate of the multigrid method for 1D convection diffusion using 2-step GMRES directly vs. 2-step GMRES inside implicit Euler smoother, $N = 128$.

c	ImpE	GMRES-2
1	0.5835	0.4870
10	0.4937	0.4870
100	0.4876	0.4870
1000	0.4871	0.4870

Table 5: Measured convergence rate of the multigrid method for 2D convection diffusion using 2-step GMRES directly vs. 2-step GMRES inside implicit Euler smoother, N=128.

unsteady problem, we obtain the system matrix $\mathbf{I} - \Delta t\mathbf{A}$. Under the assumption of a large outer CFL number, the identity matrix can be neglected and the matrix to consider is $\Delta t\mathbf{A}$ and the smoother is of the form $\mathbf{M}^S(\omega\Delta t\mathbf{A})$. To obtain the optimal coefficient for this case, we thus have to divide ω^* by Δt. This gives us a smoothing factor that is approximately the same as for the steady state and means that it is sufficient to design the smoother based on the steady state problem only.

If we instead of the pseudotime consider a CFL number, then from the condition $c_S^* = c_U^*$ on the optimal parameter, we obtain

$$c_U^* = c\Delta t_U^* \Rightarrow \Delta t_U^* = c_U^*/c = \Delta t_S^*/\Delta t.$$

This implies that in the unsteady case, the larger the outer time step, the smaller the pseudo time step has to be. However, this does not decrease the smoothing factor, as seen above.

5 NUMERICAL RESULTS

5.1 Implicit pseudo time iterations

We first test the difference between using an implicit pseudo time iteration with a certain smoother compared to using that smoother directly on the example of GMRES with 2 steps. The first test case is the one dimensional convection diffusion equation with $a = 1$, $b = 0.001$, $\Delta t = 0.5$. We now vary the pseudo time CFL number and compare the convergence rates, as seen in Table 4. The second test case is the two dimensional convection diffusion equation with convection speed $\tilde{a} = 1$ and angle $\gamma = \pi/4$, $b = 0.001$, $\Delta t = 0.5$. Again, the pseudo time CFL number c is varied and the results can be seen in table 5.

As can be seen, for about value for c of 1000, there is no essentially no difference between convergence rates, as predicted by the theory. Furthermore, the convergence rate of the method with pseudotime iteration is decidedly smaller for small c. Thus, we do not see a benefit in this class of methods.

5.2 Variants of GMRES

We now test the results of section 4.2. The first test case is the one-dimensional convection diffusion equation with $a = 1$, $b = 0.001$, $\Delta t = 0.5$. As a smoother k-step GMRES is used.

Δx	1	2	3	4
1/64	0.6643	0.3853	0.2955	0.1801
1/128	0.6979	0.427	0.3345	0.2120
1/256	0.7484	0.4522	0.3645	0.2322

Table 6: Measured convergence rate of the multigrid method for 1D convection diffusion when using k-step GMRES as a smoother.

Δx	1	2	3	4
1/64	0.8725	0.7879	0.7836	0.7097
1/128	0.8870	0.8084	0.8033	0.7333
1/256	0.9079	0.8200	0.8172	0.7467

Table 7: The $k+2$-nd root of the measured convergence rate of the multigrid method for 1D convection diffusion when using k-step GMRES as a smoother.

In Table 6, the convergence rates observed in practice of the multigrid method can be seen for different Δx and different k. As can be seen, increasing the dimension of the Krylov subspace used, respectively the degree of the polynomial, decreases the convergence rate. Furthermore, the convergence rate increases with a finer mesh width, but not terribly so.

To get a better picture of the efficiency of the smoother, we now look at the $k+2$-nd root of the convergence rate measured. As can be seen in Table 7, increasing the number of smoothing steps shows an improving trend. It is also noticeable that an even number of steps perform slightly better than an odd number.

As a second test case, we use the two-dimensional convection diffusion equation with $\tilde{a} = 1$, $\gamma = \pi/4$, $b = 0.001$, $\Delta t = 0.5$. As a smoother k-step GMRES is used. In Table 8, the convergence rates observed in practice of the multigrid method can be seen for different Δx. As can be seen, increasing the dimension of the Krylov subspace used, respectively the degree of the polynomial, decreases the convergence rate. Furthermore, the convergence rate increases with a finer mesh width, but not terribly so. Again, the $k+2$-nd root of the convergence rate measured is shown in Table 9. Now the root is approximately constant on a particular mesh, but goes up from $k = 3$ to $k = 4$. Thus, 2- and 3-step GMRES perform about the same. Since the cost per iteration in GMRES increases with due to the increasing number of scalar products, we consider 2-step GMRES to be the most promising method.

5.3 Comparison of smoothers

Now all the smoothers under consideration are compared. The first test case is the two-dimensional convection diffusion equation with $\tilde{a} = 1$, $\gamma = \pi/4$, $b = 0.001$, $\Delta t = 0.5$ and a two-level multigrid cycle. Table 10 lists the convergence rates obtained with all smoothers. We test the effect of using the smoother vs. a direct solver on the coarsest multigrid level. In Table 11, the m-th root of the convergence rates is listed; the values of m are given in Table 3. Using

Δx	1	2	3	4
1/32	0.5192	0.3853	0.2838	0.2726
1/64	0.5913	0.4506	0.3576	0.3531
1/128	0.6326	0.4870	0.4076	0.3975

Table 8: Measured convergence rate of the multigrid method for 2D convection diffusion when using k-step GMRES as a smoother.

Δx	1	2	3	4
1/32	0.8037	0.7879	0.7773	0.8052
1/64	0.8393	0.8193	0.8141	0.8407
1/128	0.8584	0.8354	0.8357	0.8575

Table 9: The $k+2$-nd root of the measured convergence rate of the multigrid method for 2D convection diffusion when using k-step GMRES as a smoother.

	direct			smoother		
Scheme	1/32	1/64	1/128	1/32	1/64	1/128
ARK4	0.5330	0.5430	0.5397	0.8034	0.8977	0.9471
ARK5	0.6473	0.6412	0.6266	0.9169	0.9580	0.9787
RK2	0.4312	0.4646	0.4789	0.7049	0.8452	0.9198
RK4	0.3505	0.4130	0.4428	0.4749	0.7129	0.8479
ImpE	0.2654	0.3541	0.4165	0.3539	0.5675	0.7608
GMRES	0.2658	0.3541	0.4158	0.3576	0.5679	0.7583
SGS	0.0875	0.2013	0.3489	0.1019	0.2854	0.5998
ILS	0.7814	0.8334	0.8494	0.7344	0.8496	0.9246

Table 10: The measured convergence rate of the 2-level multigrid method for 2D convection diffusion with $\gamma = \pi/4$ when using smoother vs. direct solve on coarsest multigrid level.

a smoother is less effective than a direct solver as a coarse-grid method. The best smoother in terms of computational effort is SGS. Explicit multistep smoothers (RK and ARK) achieve relatively poor convergence rates. The implicit line smoother achieves the slowest convergence because the lines are not aligned with the convection direction.

In the next test, the number of multigrid levels is increased to the maximum possible for each N: i.e. $\log_2(N)$. Tables 12 and 13 list the convergence rates and their m-th roots. The choice of smoother/direct solver on the coarsest grid is insignificant. Overall, convergence rates are improved by the increased number of multigrid levels. As in the two-level test, SGS is the fastest-converging smoother. However, the m-th root of convergence rate increases with N and on the finest grid RK2 has comparable values. On still finer grids SGS may not be the clear winner.

As a final test the convection angle is set to $\gamma = \pi/2$ so that the lines of ILS are aligned with the flow. The number of multigrid cycles is kept at $\log_2(N)$. Tables 14 and 15 list the convergence rates and their m-th roots respectively. Most notable is the ILS convergence rate is greatly improved at lower N. The other schemes follow the opposite trend, their convergence rates getting larger at $\gamma = \pi/2$.

6 CONCLUSIONS

We considered different smoothers in an agglomeration multigrid method for a finite volume discretization of the linear convection diffusion equation. For these, we considered different smoothers, namely explicit and additive Runge-Kutta methods, GMRES with a fixed number of steps, SGS and implicit line smoothers. We related GMRES to optimized explicit RK methods in that GMRES automatically chooses a polynomial that optimizes residual reduction over all eigenvalues, not just the ones relevant for smoothing. In the case of GMRES, we analyze and demonstrate by numerical experiments that implicit pseudo time iterations do not offer a benefit compared to using the inner method directly as a smoother. Implicit line smoothing turns out to

	direct			smoother		
Scheme	1/32	1/64	1/128	1/32	1/64	1/128
ARK4	0.8817	0.8850	0.8840	0.9572	0.9786	0.9892
ARK5	0.9301	0.9286	0.9251	0.9856	0.9929	0.9964
RK2	0.7555	0.7745	0.7824	0.8900	0.9455	0.9725
RK4	0.8109	0.8379	0.8497	0.8616	0.9346	0.9675
ImpE	0.8016	0.8411	0.8642	0.8410	0.9099	0.9555
GMRES	0.8018	0.8411	0.8639	0.8425	0.9100	0.9549
SGS	0.4985	0.6325	0.7402	0.5208	0.6989	0.8641
ILS	0.9519	0.9642	0.9679	0.9401	0.9679	0.9845

Table 11: The m-th root of the measured convergence rate of the 2-level multigrid method for 2D convection diffusion with $\gamma = \pi/4$ when using smoother vs. direct solve on coarsest multigrid level. Table 3 lists the value of m for each scheme.

	direct			smoother		
Scheme	1/32	1/64	1/128	1/32	1/64	1/128
ARK4	0.5562	0.5918	0.6143	0.5624	0.5907	0.6105
ARK5	0.6780	0.7035	0.7295	0.6876	0.7101	0.7364
RK2	0.4130	0.4275	0.4497	0.4144	0.4332	0.4558
RK4	0.2933	0.3550	0.3868	0.2931	0.3539	0.3867
ImpE	0.2716	0.3527	0.3969	0.2717	0.3527	0.3969
GMRES	0.2726	0.3533	0.3974	0.2726	0.3533	0.3974
SGS	0.0817	0.1938	0.3358	0.0818	0.1938	0.3358
ILS	0.9019	1.1877	1.4010	0.9787	1.2581	1.4712

Table 12: The measured convergence rate of the $\log_2(N)$-level multigrid method for 2D convection diffusion with $\gamma = \pi/4$ when using smoother vs. direct solve on coarsest multigrid level.

	direct			smoother		
Scheme	1/32	1/64	1/128	1/32	1/64	1/128
ARK4	0.8893	0.9004	0.9071	0.8913	0.9001	0.9060
ARK5	0.9373	0.9431	0.9488	0.9395	0.9445	0.9503
RK2	0.7447	0.7533	0.7662	0.7455	0.7566	0.7696
RK4	0.7824	0.8129	0.8270	0.7824	0.8124	0.8270
ImpE	0.8048	0.8406	0.8573	0.8048	0.8406	0.8573
GMRES	0.8052	0.8408	0.8574	0.8052	0.8408	0.8574
SGS	0.4889	0.6258	0.7321	0.4890	0.6258	0.7321
ILS	0.9796	1.0350	1.0698	0.9957	1.0470	1.0803

Table 13: The m-th root of the measured convergence rate of the $\log_2(N)$-level multigrid method for 2D convection diffusion with $\gamma = \pi/4$ when using smoother vs. direct solve on coarsest multigrid level.

	direct			smoother		
Scheme	1/32	1/64	1/128	1/32	1/64	1/128
ARK4	0.8964	0.8893	0.8539	0.8966	0.8897	0.8538
ARK5	0.9440	0.9378	0.9158	0.9443	0.9388	0.9179
RK2	0.8506	0.8469	0.7996	0.8506	0.8469	0.7995
RK4	0.7352	0.7427	0.6857	0.7352	0.7427	0.6857
ImpE	0.4314	0.5400	0.5652	0.4314	0.5400	0.5652
GMRES	0.5183	0.5719	0.5634	0.5183	0.5719	0.5634
SGS	0.1156	0.2422	0.4414	0.1156	0.2422	0.4414
ILS	0.1913	0.4768	0.7437	0.1913	0.4768	0.7437

Table 14: The measured convergence rate of the $\log_2(N)$-level multigrid method for 2D convection diffusion with $\gamma = \pi/2$ when using smoother vs. direct solve on coarsest multigrid level.

	direct			smoother		
Scheme	1/32	1/64	1/128	1/32	1/64	1/128
ARK4	0.9784	0.9768	0.9869	0.9784	0.9769	0.9869
ARK5	0.9904	0.9893	0.9855	0.9905	0.9895	0.9858
RK2	0.9475	0.9461	0.9282	0.9475	0.9461	0.9281
RK4	0.9403	0.9422	0.9273	0.9403	0.9422	0.9273
ImpE	0.8693	0.9024	0.9093	0.8693	0.9024	0.9093
GMRES	0.8963	0.9111	0.9088	0.8963	0.9111	0.9088
SGS	0.5398	0.6668	0.7917	0.5398	0.6668	0.7917
ILS	0.7183	0.8623	0.9425	0.7183	0.8623	0.9425

Table 15: The m-th root of the measured convergence rate of the $\log_2(N)$-level multigrid method for 2D convection diffusion with $\gamma = \pi/2$ when using smoother vs. direct solve on coarsest multigrid level.

be a specific block Jacobi method.

When looking at computational effort measured in terms of matrix vector products, 2-step GMRES and SGS perform best. Since the latter is not parallel, 2-step GMRES is a very interesting candidate for further study.

REFERENCES

[1] P. BIRKEN, *Optimizing Runge-Kutta smoothers for unsteady flow problems*, ETNA, 39 (2012), pp. 298–312.

[2] P. BIRKEN, *Solving nonlinear systems inside implicit time integration schemes for unsteady viscous flows*, in Recent Developments in the Numerics of Nonlinear Hyperbolic Conservation Laws, R. Ansorge, H. Bijl, A. Meister, and T. Sonar, eds., Springer, 2013, pp. 57–71.

[3] D. CAUGHEY AND A. JAMESON, *How many steps are required to solve the Euler equations of steady compressible flow: In search of a fast solution algorithm*, AIAA Paper 2001-2673, (2001).

[4] H. C. ELMAN, O. G. ERNST, AND D. P. O'LEARY, *A Multigrid Method Enhanced by Krylov Subspace Iteration for Discrete Helmholtz Equations*, SIAM J. Sci. Comput., 23 (2001), pp. 1291–1315.

[5] A. JAMESON, *Transonic flow calculations for aircraft*, in Numerical Methods in Fluid Dynamics, F. Brezzi, ed., Lecture Notes in Mathematics, Springer, 1985, pp. 156–242.

[6] A. JAMESON, W. SCHMIDT, AND E. TURKEL, *Numerical Solution of the Euler Equations by Finite Volume Methods Using Runge-Kutta Time-Stepping Schemes*, in AIAA Paper 1981-1259, 1981.

[7] S. LANGER, *Investigation and application of point implicit Runge-Kutta methods to inviscid flow problems*, Int. J. Num. Meth. Fluids, 69 (2012), pp. 332–352.

[8] S. LANGER, *Application of a line implicit method to fully coupled system of equations for turbulent flow problems*, Int. J. CFD, 27 (2013), pp. 131–150.

[9] S. LANGER AND D. LI, *Application of point implicit Runge-Kutta methods to inviscid and laminar flow problems using AUSM and AUSM + upwinding*, International Journal of Computational Fluid Dynamics, 25 (2011), pp. 255–269.

[10] S. LANGER, A. SCHWÖPPE, AND N. KROLL, *Investigation and Comparison of Implicit Smoothers Applied in Agglomeration Multigrid*, AIAA Journal, 53 (2015), pp. 2080–2096.

[11] D. MAVRIPLIS, *An assessment of linear versus nonlinear multigrid methods for unstructured mesh solvers*, Journal of Computational Physics, 175 (2002), pp. 302–325.

[12] D. J. MAVRIPLIS, *Directional Agglomeration Multigrid Techniques for High-Reynolds Number Viscous Flows*, Tech. Rep. 98-7, ICASE, 1998.

[13] B. VAN LEER, C.-H. TAI, AND K. G. POWELL, *Design of Optimally Smoothing Multi-Stage Schemes for the Euler Equations*, in AIAA 89-1933-CP, 1989, pp. 40–59.