Preconditioning of fully implicit Runge-Kutta schemes for parabolic PDEs

Gunnar A. Staff and Kent-Andre Mardal and Trygve K. Nilssen
Simula Research Laboratory
Lysaker, Norway

Abstract

Recently, the authors introduced a preconditioner for the linear systems that arise from fully implicit Runge-Kutta time stepping schemes applied to parabolic PDEs [9]. The preconditioner was a block Jacobi preconditioner, where each of the blocks were based on standard preconditioners for low-order time discretizations like implicit Euler or Crank-Nicolson. It was proven that the preconditioner is optimal with respect to the timestep and the discretization parameter in space.
In this paper we will improve the convergence by considering other preconditioners like the upper and the lower block Gauss-Seidel preconditioners, both in a left and right preconditioning setting. Finally, we improve the condition number by using a generalized Gauss-Seidel preconditioner.

1 Introduction

Since theirs introduction in 1895, the Runge-Kutta schemes have proven to be efficient for a variety of problems. For the solution of parabolic PDEs however, the fully implicit schemes are not that widespread. The majority of these problems are computed using either implicit Euler, Crank-Nicolson or some higher-order Backward Differential Formulas (BDF) scheme in time. They all result in a Helmholtz type of problem to be solved for each timestep.

Because of the quadrature of the fully implicit Runge-Kutta schemes, the system to be solved increases in dimension for increasing number of quadrature nodes. In general, for a problem where the space is discretized using m degrees of freedom, an s stage scheme will result in a system to be solved of dimension $sm \times sm$. In addition, although the ODE system matrix is symmetric and positive definite, the matrix when the Runge-Kutta scheme is employed is in general nonsymmetric and indefinite, and requires different linear solvers.

Diagonal implicit Runge-Kutta schemes (DIRK) have been introduced to solve this problem. They lead to s systems of dimension $m \times m$ to be solved, where the symmetric positive definite property is preserved. Unfortunately, this type of schemes suffer from a severe kind of order reduction for stiff problems [4, Chapter IV.15], leading to first order convergence.

Fully implicit Runge-Kutta schemes have several desirable properties such as high-order of accuracy, strong stability properties, both for linear and for some schemes also for nonlinear differential equations. They also lead to a simple implementation of adaptivity, both with respect to the timestep and the order [5]. It is therefore desirable to reduce the computational costs of solving the linear systems arising when using Runge-Kutta methods on PDEs, in order to make them competitive to multistep methods like BDF. We will do this by reusing efficient preconditioners for the Helmholtz problem in a block preconditioner for the fully implicit Runge-Kutta scheme. We will discuss both block Jacobi, and block Gauss-Seidel preconditioners, and we will also discuss both left and right preconditioning.

To the authors' knowledge the only former work on preconditioners for the fully implicit Runge-Kutta schemes applied to parabolic equations is done in Van lent and Vandewalle [8]. In [8] the time stepping system (see (7)) is preconditioned with a multigrid approximation of the fully coupled system, using a block smoother.

The benefit of the preconditioner presented in this paper is the reuse of standard preconditioner, both when it comes to code and theory. Only an efficient preconditioner for the Helmholtz problem is needed to implement our block preconditioner. We do however need a linear solver for a general non-symmetric, indefinite matrix, such as e.g. GMRES.

The remaining of this paper is organized like this: Section 2 explains the discretization of the problem, with emphasis on the resulting block structure from the Runge-Kutta discretization in time. The preconditioner is presented in Section 3, and some important properties are discussed. Section 4 presents a variety of numerical experiments, demonstrating the effectiveness of the preconditioner. In Appendix A, some optimal coefficients for the generalized lower block Gauss-Seidel preconditioner are presented.

2 Discretization of the problem

Let Ω be a bounded polygonal domain in $\mathbb{R}^d$, with d=1,2 or 3, and boundary $\partial\Omega$. We will consider a parabolic PDE on the form

$$\begin{aligned} \frac{\partial u}{\partial t} &= \Delta u + f, \quad \text{in } \Omega,\ t > 0, && (1)\\ u &= 0, \quad \text{on } \partial\Omega,\ t \geq 0 && (2)\\ u &= u^0, \quad \text{in } \Omega,\ t = 0, && (3) \end{aligned}$$

The equations (1)–(3) are discretized in space by a finite element method, which gives a $\mathbb{R}^{m\times m}$ system of ODEs to be solved,

$$\begin{aligned} I_h \frac{du_h}{dt} &= \Delta_h u_h + f_h, \quad t > 0, && (4)\\ u_h &= u_h^0, \quad t = 0, && (5) \end{aligned}$$

where I_h is the mass matrix and Δ_h is the stiffness matrix.

When the equations (4)–(5) are discretized by single stage discretization schemes such as implicit Euler, Crank-Nicolson or higher-order BDF schemes, we arrive at the following sequence of linear systems to be solved at each time step

$$(I_h + \delta t \alpha_0 \Delta_h) u_h^n = I_h u^{n-1} + \delta t(\cdots), \tag{6}$$

where δt is the time stepping parameter and α_0 is a coefficient specific for the chosen scheme. All higher-order single stage schemes result in more terms on the right hand side, leaving the left hand side unchanged (except for the α_0 parameter). This means that a Helmholtz solver has to be used to compute a single timestep.

A Runge-Kutta scheme, applied to our model problem (1)-(2), can be written on the form

$$g_i = \Delta_h \left(u_h^n + \delta t \sum_{j=1}^{s} A_{ij} g_j \right) + f_h(t^n + c_i \delta t) \tag{7}$$

$$u^{n+1} = u^n + \delta t \sum_{i=1}^{s} b_i g_i, \quad i = 1, \ldots, s \tag{8}$$

where A_{ij} are the Runge-Kutta coefficients, b_i are the quadrature weights and c_i are the quadrature nodes of the Runge-Kutta scheme, organized in the Butcher tableau

$$\begin{array}{c|c} c & A \\ \hline & b^T \end{array}.$$

To better understand the block structure arising from the Runge-Kutta system, we write the scheme on matrix form

$$\mathcal{A} = \begin{pmatrix} I_h - \delta t A_{11} \Delta_h & \cdots & -\delta t A_{1s} \Delta_h \\ \vdots & \ddots & \vdots \\ -\delta t A_{s1} \Delta_h & \cdots & I_h - \delta t A_{ss} \Delta_h \end{pmatrix}.$$

We then have to solve $\mathcal{A}\mathbf{g} = \mathbf{b}$, where $\mathbf{b}$ is the right hand side given by (7). We notice the block system, with Helmholtz problems on the diagonal, and Poisson problems on the off-diagonals. This motivates us to reuse the preconditioner for the Helmholtz problem.

We will here give a brief introduction to the different fully implicit Runge-Kutta schemes. For a more thorough description, the reader is referred to [4]. The main difference between the families of fully implicit Runge-Kutta schemes is the choice of quadrature nodes. It can either be the Gauss, the Radau or the Lobatto quadrature. The methods based on the Gauss quadrature are stable for both linear and nonlinear problems, and the order is $2s$. In addition the schemes are symplectic (see [3, Chapter II.16]). These methods will be denoted Gs, where s is the number of nodes. Note that $G1$ is the famous implicit midpoint scheme.

When choosing the Radau quadrature, we have to decide if we want the start or the endpoint to be one of the quadrature nodes. We choose the endpoint, leading to very attractive schemes for parabolic PDEs. The schemes are stable for both linear and nonlinear problems, and the order is $2s-1$. In addition the schemes are stiffly accurate [4, Chapter IV.15], which is a very attractive property when solving different PDEs. These methods will be denoted $RIIs$, where II notes that the endpoint is one of the quadrature nodes. Note that $RII1$ is the famous implicit Euler scheme.

By choosing the Lobatto quadrature nodes, we derive three subfamilies of schemes. Two of the subfamilies have an explicit step, either the first or the last quadrature node. We will not discuss these schemes here. Instead we will focus on the subfamily with only implicit quadrature points. These schemes are also stable for linear and nonlinear problems, and the order is $2s-2$. As for the RII schemes, these schemes are also stiffly accurate. We will denote them LCs, where C refers to the subfamily C.

Note that both the Gauss methods, and the RadauII methods are collocation methods. As a consequence, methods of any given order are easily constructed.

3 The Preconditioner

A general description of a preconditioned problem is

$$\mathcal{B}_L \mathcal{A} \mathcal{B}_R \mathbf{x} = \mathcal{B}_L \mathbf{b}, \quad x = \mathcal{B}_R^{-1} \mathbf{g},$$

where $\mathcal{B}_L$ and $\mathcal{B}_R$ is the left and the right preconditioner, respectively. It will be made clear from the context whether $\mathcal{B}$ is a left, or a right preconditioner, so we will from now on drop the subscript for left and right.

For $s=1$, we end up with preconditioning and solving a Helmholtz problem. Order optimal solution algorithms for this system are well known for most spatial discretization methods. The goal of this paper is to reuse such preconditioners for fully implicit Runge–Kutta schemes.

We will investigate both a block Jacobi and a block Gauss Seidel preconditioner on the form

$$\mathcal{B}_J = \begin{pmatrix} I_h - \delta t \tilde{A}_{11} \Delta_h & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & I_h - \delta t \tilde{A}_{ss} \Delta_h \end{pmatrix}^{-1} \tag{9}$$

$$\mathcal{B}_{GSL} = \begin{pmatrix} I_h - \delta t \tilde{A}_{11} \Delta_h & \cdots & 0 \\ \vdots & \ddots & \vdots \\ -\delta t \tilde{A}_{s1} \Delta_h & \cdots & I_h - \delta t \tilde{A}_{ss} \Delta_h \end{pmatrix}^{-1}, \tag{10}$$

where $\tilde{A} = A$ for now. $\tilde{A}$ will be referred to as the preconditioner coefficients. The upper triangular Gauss-Seidel (GSU) preconditioner $\mathcal{B}_{GSU}$

have the structure of the transposed of the lower triangular (GSL) preconditioner $\mathcal{B}_{GSL}$. When we write $\mathcal{B}_{GS}$, it means that it can be either lower, or upper block Gauss-Seidel preconditioner.

For lower-order discretization methods in space, multigrid and domain decomposition methods are often used as preconditioners. Such methods have been extensively studied both in theory and in practice, and it has been shown that they are order optimal with respect to the discretization parameters h and δt. When we write $\mathcal{B}$, it means exact preconditioning, meaning that each block is inverted exactly. $\tilde{\mathcal{B}}$ means that we compute a cheap approximation of $\mathcal{B}$, e.g. multigrid.

Property 1 *By using an order optimal preconditioner for the Helmholtz problem for each diagonal block, the block Jacobi preconditioner $\tilde{\mathcal{B}}_J$ will also be order optimal.*

This is proven in [9]. It can be proven in a similar way for the block Gauss-Seidel preconditioner, but we will investigate this numerically.

Property 2 *Assume that $\tilde{\mathcal{B}}$ is the approximation of the exact preconditioner $\mathcal{B}$, e.g. multigrid, and that $\mathcal{B}$ is either a block Jacobi or a block Gauss-Seidel preconditioner. Then the condition number can be bounded by*

$$\kappa(\tilde{\mathcal{B}}\mathcal{A}) \leq \kappa(\tilde{\mathcal{B}}\mathcal{B}^{-1})\kappa(\mathcal{B}\mathcal{A}) \tag{11}$$

Proof: By using a Cauchy-Schwarz like inequality valid for condition numbers, we find that

$$\begin{aligned}\kappa(\tilde{\mathcal{B}}\mathcal{A}) &= \kappa(\tilde{\mathcal{B}}\mathcal{B}^{-1}\mathcal{B}\mathcal{A})\\ &\leq \kappa(\tilde{\mathcal{B}}\mathcal{B}^{-1})\kappa(\mathcal{B}\mathcal{A})\end{aligned}$$

□

It is clear from Property 2 that the condition number using an inexact preconditioner can be bounded by the condition number using exact preconditioning multiplied by an amplification factor. This amplification factor is the condition number of the inexact preconditioner applied to the inverse of the exact preconditioner.

Assuming that we use a block Jacobi preconditioner, and that Δ_h is symmetric positive definite. Then $\tilde{\mathcal{B}}\mathcal{B}^{-1}$ is also symmetric positive definite, leading to

$$\kappa(\tilde{\mathcal{B}}\mathcal{B}^{-1}) = \frac{\max_i(\max(eig(\tilde{\mathcal{B}}_i\mathcal{B}_i^{-1})))}{\min_j(\min(eig(\tilde{\mathcal{B}}_j\mathcal{B}_j^{-1})))}$$

It is well known that $\tilde{\mathcal{B}}$ can be approximated using multigrid or domain decomposition methods.

Assume that we use a lower block Gauss-Seidel preconditioner. The inverse of a triangular matrix is still a triangular matrix. We now write

$$\mathcal{B}^{-1} = \begin{pmatrix} \hat{\mathcal{B}}_{11} & \dots & 0 \\ \vdots & \ddots & \vdots \\ \hat{\mathcal{B}}_{s1} & \dots & \hat{\mathcal{B}}_{ss} \end{pmatrix}.$$

Then we find that

$$\tilde{\mathcal{B}}\mathcal{B}^{-1} = \begin{cases} \displaystyle\sum_{k=j}^{i} \tilde{\mathcal{B}}_{ik}\hat{\mathcal{B}}_{kj} & i \geq j, \ i,j = 1,\dots,s \\ 0 & \text{else.} \end{cases}$$

Hence, the analysis is more complicated. The same argument holds for upper block Gauss-Seidel. Obviously the block Gauss-Seidel preconditioner is less robust to a poor approximation of the preconditioner, then the block Jacobi preconditioner in the case with a large number of quadrature points. We are therefore interested in investigating these amplification factors numerically when using multigrid approximation.

Property 3 *Assume that the complexity of the matrix-vector product $\Delta_h x$ scales as $\mathcal{O}(m)$, for $\Delta_h \in \mathbb{R}^{m\times m}$ and $x \in \mathbb{R}^m$. Then one iteration of our preconditioned algorithm, both the block Jacobi and the block Gauss-Seidel, scales as $\mathcal{O}(s^2 m)$, where s is the number of quadrature nodes in the chosen Runge-Kutta scheme.*

This means that the fully implicit scheme scales as the DIRK methods, but worse then the single stage schemes which scales as $\mathcal{O}(pm)$ where p is the number of steps. However, s is usually small leading to a relatively small difference. The question is if the increase in computational cost for one timestep is larger then the decrease in the required number of timesteps. This will be investigated numerically.

One benefit of the presented preconditioner is that spatial discretization technique can easily be changed. In practice people would probably be interested in using higher-order methods in space as well as in time. As long as there exists a preconditioner for the implicit Euler method, this can be reused with our methodology. Note however that the proof of Property 1 is based on a conforming finite element or spectral element discretization.

4 Results

We will use multigrid to approximate the preconditioner. All computations will be done on a domain $\Omega = (0,1)^d$, where d is the number of spatial dimensions. A sequence of meshes is constructed by uniform refinement of a 2, 2×2 or $2\times 2\times 2$ partition of the domain Ω. The preconditioner $\tilde{\mathcal{B}}$ is computed using a standard V-cycle with a symmetric Gauss-Seidel smoother. Gaussian elimination is used as the coarse grid solver. Note that we do not reach the asymptotic region for 1D multigrid preconditioning in our experiments, and the condition number may be higher in this case.

We want to find the condition number $\kappa(\tilde{\mathcal{B}}\mathcal{A})$ for left preconditioning, and $\kappa(\mathcal{A}\tilde{\mathcal{B}})$ for right preconditioning. For large problems, this can not be found exactly. It is therefore approximated by solving the linear system using Conjugate Gradient for the Normal equation (CGN). More precisely we solve

$$(\tilde{\mathcal{B}}\mathcal{A})^T\tilde{\mathcal{B}}\mathcal{A}x = (\tilde{\mathcal{B}}\mathcal{A})^T\tilde{\mathcal{B}}b$$

and approximate $\kappa(\tilde{\mathcal{B}}\mathcal{A}) = \sqrt{\kappa((\tilde{\mathcal{B}}\mathcal{A})^T\tilde{\mathcal{B}}\mathcal{A})}$. A description on how to approximate the condition number from a Conjugated Gradient method can be found in [10].

4.1 Verification of the optimality of the preconditioner using multigrid

In this experiment we verify numerically the order optimality of the block preconditioner with respect to the spatial discretization parameter h and the timestep δt, by using multigrid to approximate the blocks. This is done for the 2D problem (1)-(3) using bilinear finite elements in space and the three nodes RadauII scheme in time. First $\tilde{\mathcal{B}}_J$ is a block Jacobi preconditioner approximated by one multigrid V-cycle. The results can be found in Table 1. The order optimal behavior is confirmed with an asymptotic value of roughly 17.

$\delta t/h$	2^{-3}	2^{-4}	2^{-5}	2^{-6}	2^{-7}	2^{-8}	2^{-9}
0.1	14.3	15.2	15.7	16.2	16.6	16.8	16.9
0.05	13.4	14.9	15.4	16.0	16.4	16.7	16.9
0.02	11.1	14.1	15.1	15.6	16.1	16.5	16.8
0.01	8.49	13.0	14.7	15.3	15.9	16.3	16.7
0.005	5.71	11.2	14.1	15.1	15.6	16.1	16.5
0.002	3.03	7.84	12.6	14.5	15.3	15.8	16.3
0.001	1.99	5.17	10.6	13.8	15.0	15.5	16.1

Table 1: The condition number $\kappa(\tilde{\mathcal{B}}_J\mathcal{A})$ for the 2D problem (1)-(3) using bilinear finite elements in space, and the three nodes RadauII scheme in time. $\tilde{\mathcal{B}}_J$ is the block Jacobi preconditioner, and is approximated using one multigrid V-cycle.

In the second experiment, $\tilde{\mathcal{B}}_{GSL}$ is a lower block Gauss-Seidel preconditioner, again approximated by one multigrid V-cycle. The results can be found in Table 2. Gauss-Seidel is apparently much better then Jacobi, and the asymptotic value of the condition number is roughly 3. Again the order optimal behavior is confirmed.

4.2 Numerical investigation of the condition number when using multigrid

Property 2 states that the condition number using inexact preconditioning will be bounded by the condition number of the exact preconditioner multiplied by the condition number of inexact preconditioner applied to the inverse of the exact preconditioner. We will investigate this numerically. This is done by computing the condition number $\kappa(\mathcal{B}\mathcal{A})$ and $\kappa(\tilde{\mathcal{B}}\mathcal{A})$, where $\tilde{\mathcal{B}}$ is computed using one multigrid V-cycle. We do this for $d = 1, 2, 3$, with $h = 2^{-9}, 2^{-9}, 2^{-6}$ respectively. The exact preconditioner is only computed in the 1D case. The results using block Jacobi preconditioning can be found in Table 3, while the lower

$\delta t/h$	2^{-3}	2^{-4}	2^{-5}	2^{-6}	2^{-7}	2^{-8}	2^{-9}
0.1	2.45	2.59	2.65	2.72	2.77	2.81	2.83
0.05	2.29	2.54	2.63	2.68	2.75	2.79	2.82
0.02	1.96	2.42	2.58	2.65	2.71	2.77	2.80
0.01	1.64	2.25	2.52	2.62	2.67	2.74	2.79
0.005	1.34	1.99	2.42	2.58	2.65	2.71	2.77
0.002	1.15	1.56	2.18	2.50	2.61	2.66	2.73
0.001	1.13	1.29	1.90	2.38	2.56	2.64	2.70

Table 2: The condition number $\kappa(\tilde{\mathcal{B}}_{GSL}\mathcal{A})$ for the 2D problem (1)-(3) using bilinear finite elements in space, and the three nodes RadauII scheme in time. $\tilde{\mathcal{B}}_{GSL}$ is the lower block Gauss-Seidel preconditioner, and is approximated using one multigrid V-cycle.

block Gauss-Seidel preconditioner can be found in Table 4. The one node Gauss is included for reference.

	$\kappa(\mathcal{B}\mathcal{A})$	$\kappa(\tilde{\mathcal{B}}\mathcal{A})$		
	$1D$	$1D$	$2D$	$3D$
$G1$	1.00	1.94	1.10	1.08
$G2$	4.79	9.08	5.22	4.98
$G3$	11.8	22.0	12.7	11.9
$G4$	22.4	41.2	24.1	22.3
$G5$	37.2	67.8	40.0	36.9
$G6$	56.6	102	60.4	55.6
$RII2$	6.75	12.9	7.36	7.04
$RII3$	15.4	29.0	16.7	15.8
$RII4$	27.1	50.1	29.3	27.1
$RII5$	41.2	75.2	44.3	40.8
$RII6$	57.5	104	61.5	56.4
$LC2$	1.34	1.96	1.42	1.42
$LC3$	11.2	21.4	12.2	11.8
$LC4$	21.6	40.6	23.5	22.2

Table 3: Block Jacobi preconditioner applied to the one, two and three dimensional problem (1)-(3) with $\delta t = 0.1$ and $h = 2^{-9}, 2^{-9}, 2^{-6}$ respectively. $\tilde{\mathcal{B}}$ is computed using one multigrid V-cycle.

We notice that the increase in condition number due to the inexact preconditioning is approximately 2 in 1D, and 1.1 in 2D. For the 3D case, we are not in the asymptotic region, and the condition number is therefore some places slightly smaller then the one using exact preconditioning. We notice that the block Gauss Seidel preconditioner is in general much better then the block Jacobi preconditioner.

	$\kappa(\mathcal{BA})$	$\kappa(\tilde{\mathcal{B}}\mathcal{A})$		
	$1D$	$1D$	$2D$	$3D$
$G1$	1.00	1.94	1.10	1.08
$G2$	1.37	3.53	1.47	1.43
$G3$	2.09	6.49	2.23	2.11
$G4$	3.45	11.8	3.62	3.41
$G5$	6.57	21.9	6.99	6.38
$G6$	13.5	41.1	14.4	12.9
$RII2$	1.64	4.46	1.76	1.71
$RII3$	2.63	7.74	2.80	2.67
$RII4$	4.05	12.1	4.38	4.09
$RII5$	6.25	18.4	6.76	6.21
$RII6$	9.69	27.9	10.3	9.36
$LC2$	2.64	4.24	2.78	2.75
$LC3$	5.75	13.7	6.30	5.96
$LC4$	9.31	23.2	10.4	9.59

Table 4: Lower block Gauss-Seidel preconditioner applied to the one, two and three dimensional problem (1)-(3) with $\delta t = 0.1$ and $h = 2^{-9}, 2^{-9}, 2^{-6}$ respectively. $\tilde{\mathcal{B}}$ is computed using one multigrid V-cycle.

4.3 Comparison of left and right preconditioner for Jacobi, lower and upper Gauss-Seidel

In our fifth experiment, the difference between left and right preconditioning, for both the block Jacobi, lower block and upper block Gauss-Seidel is investigated. This is done for a 1D problem on the form (1)-(3). In space we use linear finite elements with $h = 2^{-8}$. The preconditioner is computed exact. This is done for the Radau schemes with two to six nodes. The results are shown in Table 5. From the

	$\mathcal{B}_J$		$\mathcal{B}_{GSL}$		$\mathcal{B}_{GSU}$	
	$\mathcal{BA}$	$\mathcal{AB}$	$\mathcal{BA}$	$\mathcal{AB}$	$\mathcal{BA}$	$\mathcal{AB}$
$RII2$	6.75	3.12	1.64	1.70	7.72	4.01
$RII3$	15.4	5.35	2.63	2.47	19.1	7.53
$RII4$	27.1	7.69	4.05	3.44	35.1	11.6
$RII5$	41.2	10.3	6.25	4.75	54.9	16.2
$RII6$	57.5	13.3	9.69	6.59	78.4	21.2

Table 5: The condition number for the left preconditioned system $\kappa(\mathcal{BA})$, and the right preconditioned system $\kappa(\mathcal{AB})$ for the 1D problem (1)-(3) using linear finite elements in space with $h = 2^{-8}$. $\mathcal{B}$ is the block Jacobi, lower block and upper block Gauss-Seidel, and is computed exactly.

results, we conclude that right preconditioning is generally better then left preconditioning. The difference may be more then a factor of two. We also conclude that lower block Gauss-Seidel gives the lowest condition number. Upper block Gauss-Seidel gives by far the largest condition number, which is not intuitive. The explanation to this is postponed to the next section, due to the need for some simplifying assumptions.

4.4 Finding optimal coefficients for the preconditioner

In the previous experiments we used $\tilde{A} = R(A)$, where R represents the restriction to the diagonal elements, the lower or the upper triangular part. A relevant question is if it is possible to reduce the condition number of the preconditioned system by changing $\tilde{A}$. From the previous example, we noticed that upper block Gauss-Seidel gives a larger condition number then the block Jacobi preconditioner. By choosing all the off-diagonal coefficients infinite small, we will be close to a block Jacobi preconditioner. This clearly indicates that it should be possible to find a more optimal $\tilde{A}$. In order to find these optimal coefficients, we need to understand what governs the condition number from the preconditioned system.

If we instead of solving a PDE, discretize a scalar ODE $u' = \lambda u$, we get

$$\mathcal{A} = \begin{pmatrix} 1-\delta t A_{11}\lambda & -\delta t A_{12}\lambda & \cdots & -\delta t A_{1s}\lambda \\ -\delta t A_{21}\lambda & 1-\delta t A_{22}\lambda & \cdots & -\delta t A_{2s}\lambda \\ \vdots & \vdots & \ddots & \vdots \\ -\delta t A_{s1}\lambda & -\delta t A_{s2}\lambda & \cdots & 1-\delta t A_{ss}\lambda \end{pmatrix}$$

$\mathcal{B}$ is identical, only changing A with $\tilde{A}$ and restricting it to diagonal or lower triangular. If $\delta t\lambda \gg 1$, it is obvious that

$$\kappa(\mathcal{B}\mathcal{A}) \approx \kappa(\tilde{A}^{-1}A). \tag{12}$$

For a PDE, λ will be a $n \times n$ matrix, containing n eigenvalues. If we assume that all the blocks in $\mathcal{A}$ is well preconditioned by $\mathcal{B}$, all the eigenvalues will be clustered and (12) is still a good approximation. This is tested for the 1D problem (1)-(3), using linear finite elements with $h = 2^{-8}$, and the results can be seen in Table 6.

We will now indicate why upper block Gauss-Seidel works so bad compared to block Jacobi and lower block Gauss-Seidel. Obviously we have

$$\left(\tilde{A}_{GSL}^{-1}A\right)_{ij} = \sum_{k=1}^{i} \hat{A}_{ik}A_{kj} \tag{13}$$

$$\left(\tilde{A}_{GSU}^{-1}A\right)_{ij} = \sum_{k=i}^{s} \hat{A}_{ik}A_{kj} \tag{14}$$

where $(\tilde{A}^{-1})_{ij} = \hat{A}_{ij}$. Most fully implicit Runge-Kutta schemes have large values in the lower triangular part of the coefficient matrix A, and small values in the upper triangular part.

	$\mathcal{B}_J$				$\mathcal{B}_{GSL}$			
	$\mathcal{BA}$	$\tilde{A}^{-1}A$	$\mathcal{AB}$	$A\tilde{A}^{-1}$	$\mathcal{BA}$	$\tilde{A}^{-1}A$	$\mathcal{AB}$	$A\tilde{A}^{-1}$
$RII2$	6.75	6.75	3.12	3.01	1.64	1.64	1.70	1.70
$RII3$	15.4	15.4	5.35	5.15	2.63	2.63	2.47	2.47
$RII4$	27.1	27.1	7.69	7.61	4.05	4.05	3.44	3.44
$RII5$	41.2	41.2	10.3	10.3	6.25	6.26	4.75	4.75
$RII6$	57.5	57.5	13.3	13.3	9.69	9.70	6.59	6.59

Table 6: Comparison of the condition number of the preconditioned system $\mathcal{BA}$ and the condition number of the preconditioner coefficient matrix and Runge-Kutta coefficient matrix $\tilde{A}A$. The numbers are in good agreement, motivating us to use (12) as a cheep cost-function for the optimization process.

For lower block Gauss-Seidel, $\tilde{A}_{GSL}^{-1}A$, this leads to a small number divided by a larger number in the upper right part of the matrix, while the lower part is well preconditioned. In general this leads to a small condition number.

For the upper block Gauss-Seidel, $\tilde{A}_{GSU}^{-1}A$, this do however lead to a relative large number divided by a smaller number in the lower left part of the matrix, while the upper part is well preconditioned. In general this leads to a large condition number.

Because of its bad preconditioning properties, we will discuss upper block Gauss-Seidel no more. Note that this is not a proof, but only a plausible explanation.

The same type of arguments can be used to explain why right preconditioning is generally better then left.

We will now see if it is possible to improve the conditioning number by optimizing the preconditioner coefficient matrix $\tilde{A}$. Obviously (12) is a good approximation, at least as long as the preconditioner is computed exact. We will therefore optimize the coefficients in $\tilde{A}$, given the structure from the choice of a block Jacobi scheme, or a lower block Gauss-Seidel scheme.

$$\begin{aligned} &\min_{\tilde{A}} \kappa(\tilde{A}^{-1}A), \text{ left preconditioning} \\ &\min_{\tilde{A}} \kappa(A\tilde{A}^{-1}), \text{ right preconditioning} \end{aligned} \tag{15}$$

Note that we now use a generalized block Jacobi, or block Gauss-Seidel, since $\mathcal{B}$ is no longer the block diagonal or block triangular part of $\mathcal{A}$. We use a Nelder-Mead algorithm [6] and initialize with the values from A. Note that we might not find the global optimal value by using this optimization process.

In Table 7 we see the condition numbers based on the optimized preconditioner coefficient matrix $\tilde{A}$. The difference between the optimization cost function (15) and $\kappa(\mathcal{BA})$, is minimal. $\mathcal{A}$ is constructed for the 1D heat equation (1)-(3) using linear elements with $h = 2^{-8}$.

Since the difference between left and right lower block Gauss-Seidel is relative small, and left preconditioning is the most commonly used

	$\mathcal{B}_J$				$\mathcal{B}_{GSL}$			
	$\mathcal{BA}$	$\tilde{A}^{-1}A$	$\mathcal{AB}$	$A\tilde{A}^{-1}$	$\mathcal{BA}$	$\tilde{A}^{-1}A$	$\mathcal{AB}$	$A\tilde{A}^{-1}$
$RII2$	4.01	3.76	2.72	2.27	1.21	1.00	1.24	1.00
$RII3$	7.74	7.41	4.52	4.27	1.24	1.00	1.33	1.00
$RII4$	12.9	12.6	6.82	6.77	1.45	1.39	1.49	1.03
$RII5$	20.0	18.9	9.50	9.50	1.55	1.27	1.65	1.39
$RII6$	26.2	26.2	12.4	12.4	1.91	1.72	1.76	1.54

Table 7: Comparison of the condition number of the preconditioned system $\mathcal{BA}$ and the condition number of the preconditioner coefficient matrix and Runge-Kutta coefficient matrix $\tilde{A}A$ where the preconditioner coefficient matrix is a result from the optimization process (15).

preconditioning technique, we will from now on only discuss left preconditioning.

It is now important to determine how much the condition number will grow when the exact preconditioner $\mathcal{B}$ is changed with the inexact preconditioner $\tilde{\mathcal{B}}$ based on multigrid. The results can be seen in Table 8 for the block Jacobi and the lower block Gauss-Seidel preconditioner. For block Jacobi preconditioning, we notice that the reduction in the condition number is much smaller then expected from the exact preconditioned problem. For lower block Gauss-Seidel the condition number is in some cases larger then the non optimized case. For LC3, we do not even have convergence after 3000 CGN iterations for the lower block Gauss-Seidel. To understand this we study the blocks in the preconditioner.

$$\tilde{\mathcal{B}}_i(a\Delta_h), \quad \tilde{\mathcal{B}}_i = (c\Delta_h)^{-1} \tag{16}$$

$$\tilde{\mathcal{B}}_i(I - a\Delta_h), \quad \tilde{\mathcal{B}}_i = (I - c\Delta_h)^{-1} \tag{17}$$

The condition number of (16) will not change when c changes, though the impact on the condition number of the full block matrix is more complicated. For (17) however, we can not say that the condition number will not change when c changes, considering the approximation of the preconditioner is done by multigrid.

To avoid this, we try another approach by adding the constraint

$$\operatorname{diag}(\tilde{A}) = \operatorname{diag}(A) \tag{18}$$

to the minimization problem (15). This results in an optimization only valid for block Gauss-Seidel preconditioners.

The results can be seen in Table 9. As expected, the condition number using exact preconditioning is larger for the optimization using the constraint (18), then without. But the condition number using inexact preconditioning based on multigrid is in much better agreement with the optimization results. By choosing a 6 nodes scheme, the lower block Gauss-Seidel preconditioner using one multigrid V-cycle results in a condition number of less then 2.5 for two and three dimensional problems.

	Jacobi			Gauss-Seidel		
	Optimized		Non-opt	Optimized		Non-opt
	$\kappa(\mathcal{BA})$	$\kappa(\tilde{\mathcal{B}}\mathcal{A})$	$\kappa(\tilde{\mathcal{B}}\mathcal{A})$	$\kappa(\mathcal{BA})$	$\kappa(\tilde{\mathcal{B}}\mathcal{A})$	$\kappa(\tilde{\mathcal{B}}\mathcal{A})$
$G2$	3.41	4.91	5.22	1.11	1.45	1.47
$G3$	6.97	12.0	12.7	1.20	3.46	2.23
$G4$	12.4	22.9	24.1	1.41	2.95	3.62
$G5$	19.8	38.1	40.0	1.47	3.69	6.99
$G6$	29.1	58.2	60.4	1.63	5.60	14.4
$RII2$	4.01	6.92	7.36	1.21	1.75	1.76
$RII3$	7.74	15.8	16.7	1.24	4.09	2.80
$RII4$	12.9	27.8	29.3	1.45	3.69	4.38
$RII5$	20.0	42.1	44.3	1.55	10.5	6.76
$RII6$	26.2	59.4	61.5	1.91	2.77	10.3
$LC2$	1.34	1.42	1.42	1.08	1.56	2.78
$LC3$	6.83	11.4	12.2	3.64	$--$	6.30
$LC4$	10.2	22.2	23.5	2.50	7.10	10.4

Table 8: Condition number for optimized and non optimized preconditioner coefficient matrix $\tilde{A}$ for both exact preconditioning for the 1D heat problem (1)-(3), and the 2D heat problem (1)-(3) using a multigrid approximation of the preconditioner. The optimization is not very effective when the preconditioner is approximated by multigrid. $(--)$ means that CGN did not converge after 3000 iterations.

4.5 Iteration count and timing results

Finally, we compare the wall clock time (wct) for a given test problem. We solve (1)–(3), with a source term f such that the exact solution is

$$u(x,y,t) = \sin(\omega_x x)\sin(\omega_y y)\sin(\omega_t t)$$
$$(\omega_x,\omega_y,\omega_t) = (\pi,\pi,20.5\pi),\ t \in [0,1]$$

The high number of oscillation in time is used to generate a certain degree of complexity in time. In space we discretize using linear finite elements. Both the element size h and the time–step δt is chosen such that the error is of order 10^{-5}, measured in the L^2 norm in both space and time. The preconditioner is a lower block Gauss-Seidel approximated using one multigrid V–cycle, and the linear system is solved using GMRES with restart and 5 search vectors (for RadauII 1 node we used conjugated gradients) with a stopping criterion of absolute residual equal 10^{-7}.

Notice that for GMRES the residual is only evaluated before the restart. This means that the system is possibly over–iterated, but the computational time is in general smaller due to the high cost of evaluating the residual in every iteration. We also solve the linear system using CGN.

The results are computed on a Linux machine with an Intel P4 2.8GHz processor and 1GB RAM. The result is displayed in Table 10.

The number of iterations for GMRES and CGN is comparable, but

	$\kappa(\mathcal{B}\mathcal{A})$	$\kappa(\tilde{\mathcal{B}}\mathcal{A})$		
	$1D$	$1D$	$2D$	$3D$
$G2$	1.32	2.42	1.39	1.40
$G3$	1.51	3.25	1.53	1.55
$G4$	1.59	4.11	1.65	1.67
$G5$	1.89	5.04	1.94	1.96
$G6$	2.10	6.34	2.19	2.22
$RII2$	1.56	2.97	1.65	1.67
$RII3$	1.86	3.80	1.92	1.94
$RII4$	2.10	4.65	2.12	2.17
$RII5$	2.29	5.00	2.34	2.35
$RII6$	2.25	5.33	2.30	2.32
$LC2$	1.34	1.59	1.41	1.41
$LC3$	3.00	5.46	3.15	3.18
$LC4$	4.63	8.04	4.78	4.81

Table 9: Condition number of the preconditioned system where $\tilde{A}$ is the optimal coefficients computed from the optimization problem (15), with the constraint (18).

the difference in the wall clock time is approximately a factor of 2. In our experiment, the five nodes RadauII scheme is by far the fastest. This is due to the decrease in number of required steps outweighs the increase in number of iteration for the linear solver. Implicit Euler (RII1) is very slow due to the large number of required timesteps.

Note however that no general conclusions can be drawn from this small experiment. Which scheme is the fastest depends on several properties like the regularity of the solution, the required accuracy, the implementation etc.

5 Final remarks

In this paper we have shown that the systems arising from fully implicit Runge–Kutta schemes applied to parabolic equations can be preconditioned with block diagonal and block triangular preconditioners, where the diagonal blocks are standard preconditioners developed for the backward Euler scheme. Such preconditioner are well known to be order optimal when constructed by, e.g., multigrid or domain decomposition methods.

In several numerical experiments we have demonstrated that the condition number for the preconditioned systems is bounded. We have also seen that higher-order methods are beneficial, when using efficient preconditioners, even for problems with relatively fast dynamics and modest accuracy requirements. For the six nodes RadauII scheme, the new preconditioning approach with lower block Gauss-Seidel with

		GMRES		CGN	
Method	δt	wct	k	wct	k
RII 1	1.0e-6			10^4	3
RII 2	2.0e-3	12.2	10	23.8	9.8
RII 3	1.0e-2	4.2	13	9.1	13.7
RII 4	2.5e-2	3.0	20	6.4	16.6
RII 5	5.0e-2	2.2	21	5.0	19.1

Table 10: The wall clock time (wct) measured in minutes, and the average number of iterations k for solving the 2D heat equation (1)–(3) for RadauII schemes with various number of stages. RII 1 is solved using normal CG. The discretization parameters are chosen such that the errors from the discretizations are approximately 10^{-5}. The preconditioner is approximated using one multigrid V–cycle. The higher order schemes outperforms the lower-order schemes.

optimal coefficients results in a 30 times reduction in the condition number compared to the block Jacobi preconditioner presented in the previous paper [9].

References

[1] Lawrence C. Evans. *Partial Differential Equations*. Number 19. American Mathematical Society, 1998.

[2] Wolfgang Hackbusch. *Iterative Solution of Large Sparse Systems of Equations*. Number 95. Springer Verlag, 1994.

[3] E. Hairer, S.P. Nørsett, and G. Wanner. *Solving Ordinary Differential Equations I - Nonstiff Problems*. Springer Verlag, 2nd edition, 1992.

[4] E. Hairer and G. Wanner. *Solving Ordinary Differential Equations II - Stiff and Differential-Algebraic Problems*. Springer Verlag, 2nd edition, 1996.

[5] Ernst Hairer and Gerhard Wanner. Stiff differential equations solved by Radau methods. *Journal of Computational and Applied Mathematics*, 111:93–111, 1999.

[6] J.E. Dennis Jr. and R.B. Schnabel. A View of Unconstrained Optimization. In G.L. Nemhauser, A.H.G. Rinnooy Kan, and H.J. Todd, editors, *Optimization*, pages 1–72. Elsevier, 1989.

[7] J.C. Lagarias, J.A. Reeds, M.H. Wright, and P.E. Wright. Convergence properties of the Nelder-Meas Simplex method in lower dimensions. *SIAM Journal of Optimization*, 9:112–147, 1998.

[8] J. Van lent and S. Vandewalle. Multigrid methods for implicit Runge-Kutta and boundary value method discretizations of PDEs. to appear in SIAM J. Sci. Comput, 2004.

[9] K.A. Mardal, T.K. Nilssen, and G.A. Staff. Order optimal preconditioners for implicit Runge-Kutta schemes applied to parabolic PDE's. Simula Research Laboratory, Research Report 08–2005. URL:.

[10] Yousef Saad. *Iterative Methods for Sparse Linear Systems.* SIAM, 2nd edition, 2003.

[11] V. Thomée. *Galerkin Finite Element Methods for Parabolic Problems*, volume 2nd. Springer-Verlag, 1997.

A Optimal coefficients

Here we present the optimal coefficients for the preconditioner matrix $\tilde{A}$. The coefficients are found by solving the optimization problem (15) with the constraint (18). Due to a space limitation, only 6 decimals are presented. Only the values for left preconditioned lower block Gauss-Seidel are presented, since this has been the most effective preconditioner in our experiments.

0.25	0
0.488313	0.25

Table 11: Optimal coefficients $\tilde{A}$ for the two nodes Gauss scheme

0.138888	0	0
0.224907	0.222222	0
0.143025	0.387432	0.138888

Table 12: Optimal coefficients $\tilde{A}$ for the three nodes Gauss scheme

0.086963	0	0	0
0.171390	0.163036	0	0
0.192773	0.273261	0.163036	0
0.245927	0.232027	0.273809	0.086963

Table 13: Optimal coefficients $\tilde{A}$ for the four nodes Gauss scheme

0.059231	0	0	0	0
0.094654	0.119657	0	0	0
0.118474	0.226545	0.142222	0	0
0.156695	0.244621	0.242734	0.119657	0
0.108481	0.287240	0.227631	0.206980	0.059231

Table 14: Optimal coefficients $\tilde{A}$ for the five nodes Gauss scheme

0.042831	0	0	0	0	0
0.087051	0.090190	0	0	0	0
0.112166	0.152098	0.116978	0	0	0
0.115420	0.142112	0.224669	0.116978	0	0
0.076975	0.168167	0.271509	0.217320	0.090190	0
0.081495	0.169801	0.311476	0.215085	0.145205	0.042831

Table 15: Optimal coefficients $\tilde{A}$ for the six nodes Gauss scheme

0.416666	0
0.673076	0.25

Table 16: Optimal coefficients $\tilde{A}$ for the two nodes RadauII scheme

0.196815	0	0
0.259583	0.292073	0
0.194743	0.4.1444	0.111111

Table 17: Optimal coefficients $\tilde{A}$ for the three nodes RadauII scheme

0.112999	0	0	0
0.207430	0.206892	0	0
0.280581	0.238590	0.189036	0
0.321615	0.194202	0.255668	0.0625

Table 18: Optimal coefficients $\tilde{A}$ for the four nodes RadauII scheme

0.072998	0	0	0	0
0.134217	0.146214	0	0	0
0.166967	0.191017	0.167585	0	0
0.181347	0.188433	0.174109	0.128756	0
0.168265	0.212583	0.132551	0.176719	0.04

Table 19: Optimal coefficients $\tilde{A}$ for the five nodes RadauII scheme

0.050950	0	0	0	0	0
0.090379	0.106975	0	0	0	0
0.113069	0.173695	0.136314	0	0	0
0.117967	0.202003	0.245356	0.131006	0	0
0.100245	0.235893	0.304197	0.210749	0.092430	0
0.098567	0.240736	0.329458	0.213036	0.124316	0.027777

Table 20: Optimal coefficients $\tilde{A}$ for the six nodes RadauII scheme

0.5	0
0	0.5

Table 21: Optimal coefficients $\tilde{A}$ for the two nodes LobattoC scheme

0.166666	0	0
-0.125000	0.416666	0
-0.166666	0.606060	0.166666

Table 22: Optimal coefficients $\tilde{A}$ for the three nodes LobattoC scheme

0.083333	0	0	0
-0.031715	0.25	0	0
0.070601	0.508398	0.25	0
0.132073	0.522927	0.483915	0.083333

Table 23: Optimal coefficients $\tilde{A}$ for the four nodes LobattoC scheme