ESAIM: M2AN 49 (2015) 331–347
DOI: 10.1051/m2an/2014036

ESAIM: Mathematical Modelling and Numerical Analysis
www.esaim-m2an.org

EFFICIENT PRECONDITIONING OF VARIATIONAL TIME DISCRETIZATION METHODS FOR PARABOLIC PARTIAL DIFFERENTIAL EQUATIONS [*], [**]

Stephan Weller[1] and Steffen Basting[1]

Abstract. This paper is concerned with the design of efficient preconditioners for systems arising from variational time discretization methods for parabolic partial differential equations. We consider the first order discontinuous Galerkin method (dG(1)) and the second order continuous Galerkin Petrov method (cGP(2)). The time-discrete formulation of these methods leads to a coupled 2×2 block system whose efficient solution strongly depends on efficient preconditioning strategies. The preconditioner proposed in this paper is based on a Schur complement formulation for the so called *essential* unknown. By introducing an inexact factorization of this ill-conditioned fourth order operator, we are able to circumvent complex arithmetic and prove uniform bounds for the condition number of the preconditioned system. In addition, the resulting preconditioned operator is symmetric and positive definite, therefore allowing for the usage of efficient Krylov subspace solvers such as the conjugate gradient method. For both the dG(1) and cGP(2) method, we provide optimal choices for the sole parameter of the preconditioner and deduce corresponding upper bounds for the condition number of the resulting preconditioned system. Several numerical experiments including the heat equation and a convection-diffusion example confirm the theoretical findings.

Mathematics Subject Classification. 65M12, 65M60.

Received March 19, 2014. Revised August 21, 2014
Published online February 5, 2015.

1. Introduction

For the numerical solution of parabolic partial differential equations using the finite element method, variational time discretization approaches based on a space-time formulation of the problem exhibit attractive properties. Let us just mention a few of them (with many more pointed out in [8]):

- Stability: for many of the variational time discretization approaches, and in particular for the dG(k) and cGP(k) methods which we consider in this paper, A-stability (or even strong A-stability in the case of dG(k)) [7, 10] can be proved for arbitrary degree k.

Keywords and phrases. Finite element method, time discretization, discontinuous Galerkin, preconditioning.

[*] *Parts of this work have been funded by the German Research Foundation (DFG) as part of the Priority Program (SPP) 1506 "Transport processes at fluidic interfaces".*

[**] *The authors would like to thank Friedhelm Schieweck for introducing them to the topic of variational time discretization and Eberhard Bänsch for pointing out known efficient preconditioners for fourth order problems. The authors would also like to thank the anonymous reviewers of the manuscript for their helpful remarks and suggestions.*

[1] Applied Mathematics III, Dept. of Mathematics, Cauerstr. 11, 91058 Erlangen, Germany. weller@math.fau.de

- Convergence properties: many variational time discretization schemes possess the superconvergence property, for instance dG(k) is convergent of order $2k+1$ at the nodal points [13].
- Generality: being of variational type, time discretization is treated similar to space discretization. Consequently, finite element techniques which have been studied in great detail for space discretization may be directly applied to time discretization. Let us just mention higher order or adaptive methods.

However, efficient solution strategies are not easy to design: For general variational time discretization formulations, a typical finite element discretization of the space-time domain using continuous test and trial functions will lead to a huge system of coupled equations. This can be avoided if (at least) the test space consists of piecewise discontinuous functions: allowing discontinuous test functions leads to a decoupling of the whole system into a series of smaller systems for each time step.

In this paper we are interested in the aforementioned class of methods and consider the first order discontinuous Galerkin (dG(1)) and second order continuous Galerkin Petrov (cGP(2)) methods. While for the dG(1) method both the trial and test space consist of discontinuous, piecewise first order polynomial functions in time, in the cGP(2) method continuous, piecewise second order polynomial functions are used for the trial space while the test space still consists of discontinuous, piecewise first order polynomials in time.

The time-discrete formulation for both approaches leads to a coupled 2×2 block system. Despite their similar structure, dG(1) and cGP(2) methods differ in the following properties: the dG(1) method is convergent of second order in time (with third order nodal superconvergence) while the cGP(2) is convergent of third order in time (with fourth order nodal superconvergence). However, dG(k) methods are known to be *strongly* A-stable [7] while cGP(k) methods are A-stable only [9]. The particular choice of either dG(1) or cGP(2) therefore depends on the requirements of the individual application.

For the numerical solution of the coupled 2×2 block system, various techniques were proposed: Schötzau *et al.* [11,12,14] consider the dG(k) methods for time-independent linear operators and decouple the arising system into linear problems which have the same structure as an implicit Euler discretization of the problem, but are complex valued. In [8] Richter *et al.* consider an efficient solution strategy for nonlinear parabolic problems discretized by dG(k) methods. In order to circumvent complex arithmetic, their approach is based on an (inexact) Newton method applied to the approximate factorization of the block eliminated dG(k) system. For linear equations however, this approach turns out to be quite costly. In [4], an efficient solution approach based on a multigrid preconditioned BiCGStab solver for the coupled system is presented.

Our approach in constructing a simple to implement yet efficient preconditioned method was guided by the following requirements:

(1) While the semi-discrete formulation of dG(1) and cGP(2) leads to a coupled 2×2 block system for the unknowns, essentially only one of these unknowns is needed within a time stepping procedure. It therefore might be desirable to consider a reduced formulation of the problem for this *essential* unknown.
(2) A direct factorization of the reduced formulation leads to a complex-valued problem, which we would like to avoid for the sake of efficiency as well as simple implementation.
(3) The problem under consideration is of symmetric, positive definite nature. Consequently, we would like to use methods which profit from these properties (*e.g.* the CG method) or in other words: time discretization should not destroy symmetry of the problem.
(4) The conditioning of the discretized parabolic operator under consideration should not have any influence on the condition number of the preconditioned fully discrete system, *i.e.* we expect the condition number of the preconditioned system to be uniformly bounded in space and time discretization parameters.
(5) The preconditioner should consist of "simple" components only, allowing for an efficient numerical realization. In particular, the preconditioner should profit from efficient solution strategies designed for the PDE under consideration (for instance from efficient techniques available for an implicit Euler time discretization of the problem).

In order to fulfill requirements (1) to (3), for each time discretization method we employ a particular set of basis functions for which the Schur complement formulation in the *essential* unknown defines a symmetric, positive definite operator of fourth order. While at a first glance this approach seems unfavorable due to the inherent bad conditioning of discretized fourth order operators, we proceed by applying an efficient preconditioning technique introduced for a class of fourth order problems in [2]. This preconditioner consists of the concatenation of the (slightly modified) parabolic operators under consideration and consequently can be realized very efficiently. For the resulting left-right preconditioned symmetric operator, we can prove uniform bounds for the condition number with respect to the discretization parameters.

2. Problem and notation

We will assume a parabolic equation of the following form on a given time interval $I := (0,T)$ and a domain $\Omega \subset \mathbb{R}^d, d \in \{2,3\}$:

$$\left.\begin{aligned} \partial_t u(t,x) + \nabla\cdot(-\varepsilon(x)\nabla u(t,x)) &= f(t,x) && \text{in } I\times\Omega \\ u &= 0 && \text{on } I\times\partial\Omega \\ u(0) &= u_0 && \text{in } \Omega \end{aligned}\right\} \tag{2.1}$$

where the matrix $\varepsilon = \varepsilon(x)$ is symmetric and uniformly positive definite.

For the ease of notation we will restrict ourselves to homogeneous Dirichlet boundary conditions, the extension to more general boundary conditions is straightforward.

Remark 2.1. For the analysis we assume a diffusion coefficient $\varepsilon = \varepsilon(x)$ that is constant in time. A more general variant $\varepsilon = \varepsilon(t,x)$ can also be treated, however the preconditioning techniques need to be modified in this case (see also the remark at the end of Chap. 4).

We will use a weak formulation of equation (2.1) in both space and time. For that purpose, we consider the space of square integrable functions $L^2(\Omega)$, the Sobolev space of once weakly differentiable functions $\mathbb{V} := H_0^1(\Omega) = \{u \in H^1(\Omega) :\ u_{|\partial\Omega} = 0\}$, and its dual space $\mathbb{V}^* = H^{-1}(\Omega)$. We now define the bilinear form $a : \mathbb{V}\times\mathbb{V} \to \mathbb{R}$ by

$$a(u,v) := \int_\Omega \varepsilon(x)\nabla u\cdot\nabla v\,\mathrm{d}x. \tag{2.2}$$

For the weak formulation in time, we define the function space

$$X := \{v \in L^2((0,T),\mathbb{V}) :\ \partial_t v \in L^2((0,T),\mathbb{V}^*)\}. \tag{2.3}$$

We can now restate (2.1) in a weak setting as follows: *Find $u \in X$ such that*

$$\int_0^T \langle\partial_t u, v\rangle\,\mathrm{d}t + \int_0^T a(u,v)\,\mathrm{d}t + (u(0),v(0)) = \int_0^T \langle f, v\rangle\,\mathrm{d}t + (u_0, v(0)) \tag{2.4}$$

for all $v \in X$. Here, $(\cdot,\cdot)$ denotes the L^2-inner product and $\langle\cdot,\cdot\rangle$ is the duality pairing on $\mathbb{V}$. We require an initial value $u_0 \in \mathbb{V}$ and right-hand side $f \in L^2((0,T),\mathbb{V}^*)$. Note that under these assumptions, existence and uniqueness of a weak solution can be proved (see *e.g.* [5]).

3. Variational time discretization for parabolic partial differential equations

In order to discretize equation (2.4) in time, we divide the time interval $I = (0,T)$ in N subintervals $I_n = (t_{n-1}, t_n]$, where $0 = t_0 < t_1 < \cdots < t_N = T$. We define the time step size as $\tau_n := t_n - t_{n-1}$.

The general idea of variational time discretization is to approximate the function $u : I \to \mathbb{V}$ by a piecewise polynomial function u_τ where $u_{\tau|I_n} \in \mathbb{P}_k(I_n,\mathbb{V})$. To this end, we define the discontinuous space

$$\mathbb{P}_k^{\mathrm{dc}} := \{v \in L^2((0,T),\mathbb{V}) :\ v_{|I_n} \in \mathbb{P}_k(I_n,\mathbb{V})\ \forall n = 1,\ldots,N\} \tag{3.1}$$

and its continuous counterpart

$$\mathbb{P}_k^{\mathrm{c}} := \{v \in C^0([0,T],\mathbb{V}) : \; v_{|I_n} \in \mathbb{P}_k(I_n,\mathbb{V}) \; \forall n = 1,\dots,N\}, \tag{3.2}$$

where

$$\mathbb{P}_k(I_n,\mathbb{V}) := \{v : I_n \to \mathbb{V} : \; v(t) = \sum_{j=0}^{k} w_j t^j \; \forall t \in I_n, w_j \in \mathbb{V} \; \forall j = 0,\dots,k\}$$

denotes the space of $\mathbb{V}$-valued polynomials of order k in time. For functions $v \in \mathbb{P}_k^{\mathrm{dc}}$, we define the jump at t_n as follows:

$$v_n^- := \lim_{t \nearrow t_n} v_\tau(t), \quad v_n^+ := \lim_{t \searrow t_n} v_\tau(t), \quad [v_\tau]_n := v_n^+ - v_n^-$$

where v_0^- is to be understood as $v_\tau(0) := v_0$, *i.e.* some prescribed initial value.

For discontinuous Galerkin methods, the trial space as well as the test space are chosen as $\mathbb{P}_k^{\mathrm{dc}}$; for continuous Galerkin–Petrov methods the trial space is $\mathbb{P}_k^{\mathrm{c}}$, while the test space is $\mathbb{P}_{k-1}^{\mathrm{dc}}$.

3.1. Discontinuous Galerkin methods

The discontinuous Galerkin method of order $k \geq 0$ (dG(k)) reads:
For given $u_\tau(0) = u_0$ and $f \in L^2((0,T),\mathbb{V}^*)$, find $u_\tau \in \mathbb{P}_k^{\mathrm{dc}}$, such that

$$\int_{I_n} (\partial_t u_\tau, v) + a(u_\tau, v) \; \mathrm{d}t + ([u_\tau]_{n-1}, v_{n-1}^+) = \int_{I_n} (f,v) \; \mathrm{d}t \qquad \forall v \in \mathbb{P}_k^{\mathrm{dc}}, \quad 1 \leq n \leq N. \tag{3.3}$$

This method is known to be strongly A-stable (L-stable) (Chap. 4.1.3, Lems. 3 and 5 in [10]). While the method is of second order when measured in $\|\cdot\|_{L^2((0,T),\mathbb{V})}$, it is super convergent of order $2k+1$ when only considering the solution at the nodal values $u_\tau(t_n)$ (Thm. 12.3, p. 211 in [13]).

To derive an efficient solution strategy which still profits from the superconvergence property, we only consider $k = 1$ in the following. To this end, we need to choose a basis of $\mathbb{P}_1^{\mathrm{dc}}$.

Following [10], we remark that u_τ may be written in terms of a Lagrange basis $\{\varphi_{n,j} \in \mathbb{P}_k(I_n,\mathbb{R}), j = 1,\dots,k+1\}$

$$u_\tau(t) = \sum_{j=1}^{k+1} U_n^j \varphi_{n,j}(t) \quad \forall t \in I_n \tag{3.4}$$

where $U_n^j \in \mathbb{V}$. Correspondingly, any test function $v_j \in \mathbb{P}_k^{\mathrm{dc}}$ can be decomposed on the time interval I_n as a sum of terms of the form:

$$v(x)\varphi(t) \tag{3.5}$$

where $v \in \mathbb{V}$ is constant in time and $\varphi \in \mathbb{P}_k(I_n,\mathbb{R})$. The latter space has a finite dimensional basis $\varphi_{n,j}, j = 1,\dots,k+1$.

We use basis functions with $\varphi_{n,j}(t_{n,i}) = \delta_{i,j}$, where $t_{n,i}$ correspond to the nodes of the $k+1$-point right-sided Gauß–Radau formula. For $k = 1$, those are $t_{n,1} = t_{n-1} + \tau_n/3; t_{n,2} = t_{n-1} + \tau_n = t_n$ and the basis functions are defined via the unit interval as $\varphi_{n,1}(t) = \tilde{\varphi}_1(\tilde{t}) = \frac{3}{2}(1-\tilde{t})$ and $\varphi_{n,2}(t) = \tilde{\varphi}_2(\tilde{t}) = \frac{3}{2}(\tilde{t} - \frac{1}{3})$ where $t = t_{n-1} + \tau_n \tilde{t}$ for $\tilde{t} \in [0,1]$.

Using these basis functions as test and trial functions, and evaluating the integrals in equation (3.3), we derive the following formulation of our problem for each time interval I_n:

For given $U_{n-1}^2 \in \mathbb{V}$ from the previous time step or initial data, find $U_n^1, U_n^2 \in \mathbb{V}$ such that the following equations hold for all $v \in \mathbb{V}$:

$$\begin{aligned} \frac{3}{4}(U_n^1, v) + \frac{1}{4}(U_n^2, v) + \frac{\tau_n}{2} a(U_n^1, v) &= (U_{n-1}^2, v) + \int_{I_n} (1 - \tilde{t}) \langle f, v \rangle \ \mathrm{d}t \\ -\frac{9}{4}(U_n^1, v) + \frac{5}{4}(U_n^2, v) + \frac{\tau_n}{2} a(U_n^2, v) &= -(U_{n-1}^2, v) + \int_{I_n} 3(\tilde{t} - \frac{1}{3}) \langle f, v \rangle \ \mathrm{d}t. \end{aligned} \tag{3.6}$$

Remark 3.1. Note that this particular choice of basis functions leads to a formulation in which the term $a(U_n^2, v)$ vanishes in the first equation and $a(U_n^1, v)$ in the second. In other words, the equations are only coupled by the occurrence of the terms (U_n^i, v) in both equations. This fact will later be exploited by our preconditioner.

3.2. Continuous Galerkin–Petrov methods

The continuous Galerkin–Petrov method of order $k \geq 1$ (cGP(k)) reads:

For given $u_\tau(0) = u_0$ and $f \in L^2((0, T), \mathbb{V}^*)$, find $u_\tau \in \mathbb{P}_k^{\mathrm{c}}$, such that

$$\int_{I_n} (\partial_t u_\tau, v) + a(u_\tau, v) \ \mathrm{d}t = \int_{I_n} (f, v) \ \mathrm{d}t \qquad \forall v \in \mathbb{P}_{k-1}^{\mathrm{dc}}, 1 \leq n \leq N. \tag{3.7}$$

In contrast to the dG(k) method, cGP(k) is known to be A-stable but not strongly A-stable (Thm. 5.1 in [9]). However, for the case of $k = 2$, which we consider in this paper, the convergence properties are more favorable: cGP(2) is convergent of order 3 in the L^2-sense and superconvergent of order 4 at the nodal points (Thm. 4.2 in [1]).

As before, we will use a Lagrange basis for trial and test functions on I_n, and get

$$u_\tau(t) = \sum_{j=0}^{k} U_n^j \varphi_{n,j}(t) \qquad \forall t \in I_n \tag{3.8}$$

$$v_j(t, x) := v(x) \psi_{n,j}(t) \qquad j = 1, \ldots, k \tag{3.9}$$

where $U_n^j, v \in \mathbb{V}$ and $v_j, j = 1, \ldots, k$ form a basis of $\mathbb{P}_{k-1}^{\mathrm{dc}}$, φ are Lagrange functions of order k with Lagrange nodes corresponding to the k-point Gauß–Lobatto formula. For $k = 2$, the Lagrange nodes are $t_{n,0} = t_{n-1}, t_{n,1} = t_{n-1} + \tau_n/2, t_{n,2} = t_n$. $\psi_{n,j}$ are polynomial functions of order $k - 1$ such that $\int_{I_n} \varphi_{n,i} \psi_{n,j} = \delta_{ij}$; $i, j = 1, \ldots, k$.

The continuity of $u_\tau(t)$ is enforced by the additional condition

$$U_n^k = U_{n+1}^0, \ n = 0, \ldots, N-1 \tag{3.10}$$

with $U_0^0 := u_\tau(0) = u_0 \in \mathbb{V}$.

We derive the following problem:

For given $U_n^0 = U_{n-1}^2 \in \mathbb{V}$ from the previous time step or initial data, find $U_n^1, U_n^2 \in \mathbb{V}$ such that the following equations hold for all $v \in \mathbb{V}$:

$$\begin{aligned} (U_n^1, v) + \frac{1}{4}(U_n^2, v) + \frac{\tau_n}{2} a(U_n^1, v) &= \frac{5}{4}(U_n^0, v) - \frac{\tau_n}{4} a(U_n^0, v) + \frac{3}{2} \int_{I_n} (1 - \tilde{t}) \langle f, v \rangle \ \mathrm{d}t \\ -4(U_n^1, v) + 2(U_n^2, v) + \frac{\tau_n}{2} a(U_n^2, v) &= -2(U_n^0, v) - \frac{\tau_n}{2} a(U_n^0, v) + \frac{1}{2} \int_{I_n} (2\tilde{t} - 1) \langle f, v \rangle \ \mathrm{d}t. \end{aligned} \tag{3.11}$$

Note that the structure of this system is essentially the same as in the dG(1) case, (3.6).

3.3. Space discretization and Schur-complement formulation

We will first describe both dG(1) and cGP(2) in a common setting. To this end, define the abstract problem: *For $n = 1, \ldots, N$, find $U_n^1, U_n^2 \in \mathbb{V}$, such that the following equations hold:*

$$\begin{aligned} \mu_1(U_n^1, v) + \alpha(U_n^2, v) + \frac{\tau_n}{2} a(U_n^1, v) &= f_n(v) \forall v \in \mathbb{V} \\ -\beta(U_n^1, v) + \mu_2(U_n^2, v) + \frac{\tau_n}{2} a(U_n^2, v) &= g_n(v) \forall v \in \mathbb{V} \end{aligned} \tag{3.12}$$

where $\alpha, \beta, \mu_i \in \mathbb{R}_+, f_n, g_n \in \mathbb{V}^*$ are given as follows:

$$\begin{array}{l} \hline \text{dG}(1) \\ \hline \mu_1 = \frac{3}{4} \quad \mu_2 = \frac{5}{4} \quad \alpha = \frac{1}{4} \quad \beta = \frac{9}{4} \\ f_n(v) = (U_{n-1}^2, v) + \int_{I_n} (1 - \tilde{t}) \langle f, v \rangle \, \mathrm{d}t \\ g_n(v) = -(U_{n-1}^2, v) + \int_{I_n} 3(\tilde{t} - \frac{1}{3}) \langle f, v \rangle \, \mathrm{d}t \\ \hline \text{cGP}(2) \\ \hline \mu_1 = 1 \quad \mu_2 = 2 \quad \alpha = \frac{1}{4} \quad \beta = 4 \\ f_n(v) = \frac{5}{4}(U_n^0, v) + \frac{\tau_n}{4} a(U_n^0, v) + \frac{3}{2} \int_{I_n} (1 - \tilde{t}) \langle f, v \rangle \, \mathrm{d}t \\ g_n(v) = -2(U_n^0, v) - \frac{\tau_n}{2} a(U_n^0, v) + \frac{1}{2} \int_{I_n} (2\tilde{t} - 1) \langle f, v \rangle \, \mathrm{d}t \\ \hline \end{array} \tag{3.13}$$

Let us denote by $\mathbb{V}_h \subset \mathbb{V}$ a conforming finite element space. On $\mathbb{V}_h$, we introduce the operators $A_h, A_{h,i} : \mathbb{V}_h \to \mathbb{V}_h$ for $i = 1, 2$ by:

$$(A_h U_h, v_h) = a(U_h, v_h) \quad \forall U_h, v_h \in \mathbb{V}_h \tag{3.14}$$

$$(A_{h,i} U_h, v_h) = \mu_i(U_h, v_h) + \frac{\tau_n}{2} a(U_h, v_h) \quad \forall U_h, v_h \in \mathbb{V}_h \tag{3.15}$$

and the right-hand sides:

$$(F_h, v_h) = f_n(v_h) \quad \forall v_h \in \mathbb{V}_h \tag{3.16}$$

$$(G_h, v_h) = g_n(v_h) \quad \forall v_h \in \mathbb{V}_h. \tag{3.17}$$

On each time interval I_n, we can restate problem (3.12) as a matrix valued problem on $\mathbb{V}_h$ as follows:

$$\begin{pmatrix} A_{h,1} & \alpha I \\ -\beta I & A_{h,2} \end{pmatrix} \begin{pmatrix} U_h^1 \\ U_h^2 \end{pmatrix} = \begin{pmatrix} F_h \\ G_h \end{pmatrix}. \tag{3.18}$$

We first note from (3.12), (3.13) that during the time-stepping procedure, the unknown U_h^1 is not necessarily needed at the next time interval. Thus, we reduce (3.18) to a Schur-complement formulation for the *essential* unknown U_h^2:
Solve

$$S_h U_h^2 = A_{h,1} G_h + \beta F_h, \text{ where } S_h := \alpha\beta I + A_{h,1} A_{h,2}. \tag{3.19}$$

Remark 3.2.

- Note that the operators $A_{h,1}, A_{h,2}$ defined in (3.15) share the same eigenvectors, and therefore $A_{h,1}A_{h,2} = A_{h,2}A_{h,1}$. As a consequence, S_h is a symmetric operator, while the original formulation is in general not symmetric.
- S_h is a fourth order operator, *i.e.* the condition of problem (3.19) is even worse than the condition of (3.18).
- The efficient solution of (3.19) using iterative solvers will only be possible with a suitable preconditioner which will be constructed in the next section.

4. Efficient preconditioning of the Schur-complement formulation

In [2], a preconditioner for a class of fourth order problems was introduced and analyzed. The main ingredient is an inexact factorization of the Schur complement S, for which uniform bounds can be proved. Note that an exact factorization is also possible, but leads to complex-valued equations, as was pointed out in [8].

To simplify notation in this section, we omit the subindex h indicating the discrete space level of the objects and write τ instead of τ_n.

As a motivation, let us first consider the (symmetric) left-right preconditioned operator $A_1^{-1}SA_1^{-1}$. Note that the spectrum $\sigma(A_i)$ is given by:

$$\sigma(A_i) = \left\{\mu_i + \frac{\tau}{2}\lambda : \lambda \in \sigma(A)\right\} \subset (\mu_1, \infty). \tag{4.1}$$

Therefore, for fixed $\lambda_1 \in \sigma(A_1)$, the corresponding eigenvalue of the preconditioned operator $A_1^{-1}SA_1^{-1}$ reads as follows:

$$\tilde{\lambda}(\lambda_1) = \frac{\alpha\beta + \lambda_1^2 + \lambda_1(\mu_2 - \mu_1)}{\lambda_1^2} \tag{4.2}$$

and since $\tilde{\lambda}$ is monotonously decreasing in λ_1, we have

$$\sup_{\xi\in\sigma(A_1)} \tilde{\lambda}(\xi) \le \tilde{\lambda}(\mu_1) = \frac{\alpha\beta + \mu_1\mu_2}{\mu_1^2} \tag{4.3}$$

$$\inf_{\xi\in\sigma(A_1)} \tilde{\lambda}(\xi) \ge \lim_{\lambda_1\to\infty} \tilde{\lambda}(\lambda_1) = 1 \tag{4.4}$$

and consequently $\sigma(A_1^{-1}SA_1^{-1}) \subset (1, \frac{\alpha\beta+\mu_1\mu_2}{\mu_1^2})$. Hence, the condition number of the preconditioned operator is bounded by

$$\kappa(A_1^{-1}SA_1^{-1}) \le \frac{\alpha\beta + \mu_1\mu_2}{\mu_1^2}. \tag{4.5}$$

For dG(1), this yields $\kappa \le \frac{8}{3}$ and for cGP(2) $\kappa \le 3$, *i.e.* this preconditioned operator is already uniformly bounded with respect to space and time discretization parameters. However, this bound can be easily improved in the following way: instead of taking A_1^{-1} as both left- and right-preconditioner, we choose $A_* := \mu_* I + \frac{\tau}{2}A$, with μ_* to be determined. Note that $A_* = A_1 + (\mu_* - \mu_1)I$ and consequently its spectrum is shifted by a constant $\mu_* - \mu_1$ compared to A_1:

$$\sigma(A_*) = \{\xi + (\mu_* - \mu_1) : \xi \in \sigma(A_1)\}. \tag{4.6}$$

The eigenvalues of $S^* := A_*^{-1}SA_*^{-1}$ are given by

$$\lambda_*(\mu_*, \lambda_1) = \frac{\alpha\beta + \lambda_1^2 + \lambda_1(\mu_2 - \mu_1)}{(\lambda_1 + (\mu_* - \mu_1))^2}, \quad \lambda_1 \in \sigma(A_1). \tag{4.7}$$

Note that since all operators commute, S^* is symmetric and positive definite, which makes it well-suited for the method of conjugate gradients (CG).

Now we determine the optimal parameter μ_{opt} for each method, such that the condition number of the preconditioned operator is minimized, *i.e.* we solve

$$\mu_{\text{opt}} = \underset{\mu_* \in \mathbb{R}^+}{\operatorname{argmin}} \, \kappa(S^*) = \underset{\mu_* \in \mathbb{R}^+}{\operatorname{argmin}} \frac{\sup_{\lambda_1 \in \sigma(A_1)} \lambda_*(\mu_*, \lambda_1)}{\inf_{\lambda_1 \in \sigma(A_1)} \lambda_*(\mu_*, \lambda_1)}. \tag{4.8}$$

A tedious, but elementary computation yields the following values:

	dG(1)	cGP(2)
μ_{opt}	$\frac{\sqrt{6}}{2}$	$\sqrt{3}$
$\kappa(S^{\text{opt}})$	$\leq 6 - 2\sqrt{6} \approx 1.10$	$\leq 8 - 4\sqrt{3} \approx 1.07.$

In the following we use the notation $A_{\text{opt}} := \mu_{\text{opt}} I + \frac{\tau}{2} A$ for the optimal preconditioner (where for the sake of brevity we do not distinguish between μ_{opt} for dG(1) and cGP(2)).

Remark 4.1.

- So far we have assumed A to be symmetric and positive definite. Numerical experiments suggest that the preconditioner also works if this condition is violated. In particular, for a convection-diffusion problem, the method works even if the diffusion part is very small (see Sect. 6.4).
- In the case of a time dependent diffusion coefficient $\varepsilon(t, x)$ the induced operators A_1, A_2 no longer commute. In this case, the discrete Schur complement operator $S = \alpha\beta I + A_1 A_2$ is no longer symmetric, and the analysis presented here for the symmetrically preconditioned operator does not apply. However, following [2], the application of a non-symmetric left-right preconditioner of the form $A_1^{-1} S A_2^{-1}$ might form a possible extension.
  Also, the time dependent coefficient will change the structure of equations (3.6) and (3.11) as the terms $a(U_n^2, v), a(U_n^1, v)$ will no longer vanish in the first and second equation, correspondingly. However, if the time dependent term is approximated by the same Radau/Lobatto rules used for the basis functions, this difficulty vanishes. This approach is comparable to approximating a nonlinearity in the underlying PDE by numerical quadrature, which does not change the order of the method, as has been demonstrated in [10].

5. Numerical realization

In this chapter we elaborate on the choice of a concrete finite element space and comment on our implementation.

5.1. Choice of finite element space

We will now go into some detail about the choice of our FE space $\mathbb{V}_h$. Note that for the estimates in the last chapter, we did not require any special kind of space discretization, conformity ($\mathbb{V}_h \subset \mathbb{V}$) being the only exception. This makes our preconditioner suitable for a broad class of finite element discretizations.

In our numerical realization, we chose a polynomial ansatz on simplicial elements. To this end, let $\mathcal{T}_h$ be a shape regular triangulation, and define the space of continuous, piecewise polynomial functions on Ω:

$$\mathbb{V}_h = \mathbb{V}_{h,k} := \{v \in C^0(\Omega) : \; v_{|T} \in \mathbb{P}_k(T, \mathbb{R}) \; \forall T \in \mathcal{T}_h\}. \tag{5.1}$$

For a basis $\varphi_i, i = 1, \ldots, n_{\text{dof}}$ of $\mathbb{V}_{h,k}$, define the mass matrix, stiffness matrix, and right-hand sides by

$$\mathbf{M}_{ij} = (\varphi_j, \varphi_i) \qquad i, j = 1, \ldots, n_{\text{dof}} \tag{5.2}$$

$$\mathbf{A}_{ij} = a(\varphi_j, \varphi_i) \qquad i, j = 1, \ldots, n_{\text{dof}} \tag{5.3}$$

$$\mathbf{F}_i = (F_h, \varphi_i) \qquad i = 1, \ldots, n_{\text{dof}} \tag{5.4}$$

$$\mathbf{G}_i = (G_h, \varphi_i) \qquad i = 1, \ldots, n_{\text{dof}}. \tag{5.5}$$

With the above defined matrices $\mathbf{M}, \mathbf{A}$ and vectors $\mathbf{F}, \mathbf{G}$ it holds:

$$(IU_h, \varphi_l) = (U_h, \varphi_l) = (\mathbf{M}U)_l \tag{5.6}$$

$$(A_h U_h, \varphi_l) = a(U_h, \varphi_l) = (\mathbf{A}U)_l \tag{5.7}$$

$$(A_{h,i} U_h, \varphi_l) = \mu_i (U_h, \varphi_l) + \frac{\tau}{2} a(U_h, \varphi_l) = \Big((\mu_i \mathbf{M} + \frac{\tau}{2}\mathbf{A})U\Big)_l =: (\mathbf{A}_i U)_l \tag{5.8}$$

$$(F_h, \varphi_l) = \mathbf{F}_l \tag{5.9}$$

$$(G_h, \varphi_l) = \mathbf{G}_l \tag{5.10}$$

for $l = 1, \ldots, n_{\text{dof}}$, where $U \in \mathbb{R}^{n_{\text{dof}}}, U = (U_i)_{i=1,\ldots,n_{\text{dof}}}$ is the coefficient vector of $U_h \in \mathbb{V}_{h,k}$.

To derive the matrix-vector equation that corresponds to the operator equation (3.19) in $\mathbb{V}_h$ we explain at first the isomorphism $r_m : \mathcal{L}(\mathbb{V}_h, \mathbb{V}_h) \to \mathbb{R}^{N \times N}, N = n_{\text{dof}}$, that assigns to a linear operator $A : \mathbb{V}_h \to \mathbb{V}_h$ its matrix representation $r_m(A) \in \mathbb{R}^{N \times N}$ in the following way (see [2]): let $r_v : \mathbb{V}_h \to \mathbb{R}^N$ denote the finite element isomorphism that assigns to a function $U_h \in \mathbb{V}_h$ its nodal vector representation $U = r_v(U_h) \in \mathbb{R}^N$ as described above. Then, the matrix representation $r_m(A) \in \mathbb{R}^{N \times N}$ is defined by the property

$$r_v(AU_h) = r_m(A) r_v(U_h) \qquad \forall U_h \in \mathbb{V}_h.$$

If we define as above the matrix $\mathbf{M} \in \mathbb{R}^{N \times N}$ and the matrix $\mathbf{A} \in \mathbb{R}^{N \times N}$ assigned to A by means of $\mathbf{A}_{kj} := (A\varphi_j, \varphi_k)$ for all $k, j = 1, \ldots, N$, we can show that $r_m(A) = \mathbf{M}^{-1}\mathbf{A}$. For two operators A_1, A_2 with corresponding matrices $\mathbf{A}_1, \mathbf{A}_2$ we get $r_m(A_1 A_2) = \mathbf{M}^{-1}\mathbf{A}_1\mathbf{M}^{-1}\mathbf{A}_2$. Now, using the fact that $r_v(G_h) = \mathbf{M}^{-1}\mathbf{G}$ and the analog for F_h, we get that the operator equation (3.19) for $U_h \in \mathbb{V}_h$ is equivalent to the following matrix-vector equation for the nodal vector $U = r_v(U_h) \in \mathbb{R}^N$:

$$\left(\mathbf{I} + \frac{1}{\alpha}\mathbf{M}^{-1}\mathbf{A}_1 \frac{1}{\beta}\mathbf{M}^{-1}\mathbf{A}_2\right) U = \frac{1}{\alpha}\mathbf{M}^{-1}\mathbf{A}_1 \frac{1}{\beta}\mathbf{M}^{-1}\mathbf{G} + \frac{1}{\alpha}\mathbf{M}^{-1}\mathbf{F}. \tag{5.11}$$

The preconditioner A_{opt} has the matrix representation

$$\mathbf{I} + \frac{\tau}{2\mu_{\text{opt}}}\mathbf{M}^{-1}\mathbf{A}. \tag{5.12}$$

Note that an operator A_h has the same eigenvalues as its matrix representation $r_m(A_h)$ such that the results of Section 4 on the preconditioning directly can be applied to the iterative solution of the corresponding matrix-vector equation (5.11).

The solution of equation (5.11) can be done using the preconditioned Conjugate Gradient method. A pseudocode variant containing the operators from our setting is shown in Algorithm 1. Note that we have removed all unnecessary explicit inversions of $\mathbf{M}$ by an appropriate scaling of equations (5.11) and (5.12).

Remark 5.1. The main numerical effort in Algorithm 1 lies in the evaluation of the preconditioner $(\mu_{\text{opt}}\mathbf{M} + \frac{\tau}{2}\mathbf{A})^{-1}$. This evaluation corresponds to solving one step of the implicit Euler method with time step size $\frac{\tau}{2\mu_{\text{opt}}}$. This problem is very well studied and can be solved efficiently with many methods, *e.g.* Multigrid methods, Krylov subspace methods etc. Additionally, one inversion of the mass matrix $\mathbf{M}$ is necessary per iteration (in the application of the Schur-complement operator). This can be realized with any existing solver as above.

5.2. Implementation

The implementation was carried out with the help of the finite element framework FEniCS [6]. Due to the high level of abstraction in this framework, we were able to perform numerical experiments with polynomial bases of different degrees (1 to 4) with nearly no additional implementational effort.

For the application of the preconditioner, a Conjugate Gradient method preconditioned by the AMG preconditioner from the "hypre" library [3] was used. For the outer CG iteration (Algorithm 1), we used a relatively small (absolute) tolerance of tol = 1e-10, and a somewhat smaller tolerance of 1e-12 for the inner solvers.

Algorithm 1. Uniformly, symmetrically preconditioned Conjugate Gradient method.

```
$p \leftarrow p_0$                                                                  ▷ Initial guess $p_0$
$b \leftarrow \left(\mathbf{A}_1\mathbf{M}^{-1}\mathbf{G} + \beta\mathbf{F}\right)$                                     ▷ Compute right-hand side
$r \leftarrow \left(\alpha\beta\mathbf{M} + \mathbf{A}_1\mathbf{M}^{-1}\mathbf{A}_2\right) p - b$                              ▷ Compute initial residual
$d \leftarrow (\mu_{\text{opt}}\mathbf{M} + \frac{\tau}{2}\mathbf{A})^{-1}\mathbf{M}(\mu_{\text{opt}}\mathbf{M} + \frac{\tau}{2}\mathbf{A})^{-1}r$        ▷ Compute preconditioned residual
$\delta \leftarrow \langle r, d\rangle$
$\delta_{\text{new}} \leftarrow \delta$
loop
    $q \leftarrow \left(\alpha\beta\mathbf{M} + \mathbf{A}_1\mathbf{M}^{-1}\mathbf{A}_2\right) d$                                ▷ Apply operator
    $\rho \leftarrow \langle d, q\rangle$
    return if $\rho = 0$
    $\rho \leftarrow \frac{\delta}{\rho}$
    $p \leftarrow p - \rho d$                                                       ▷ Update solution
    $r \leftarrow r - \rho q$                                                       ▷ Update residual
    $q \leftarrow (\mu_{\text{opt}}\mathbf{M} + \frac{\tau}{2}\mathbf{A})^{-1}\mathbf{M}(\mu_{\text{opt}}\mathbf{M} + \frac{\tau}{2}\mathbf{A})^{-1}r$       ▷ Apply preconditioner
    $\delta_{\text{new}} \leftarrow \langle r, q\rangle$
    return if $\sqrt{\delta_{\text{new}}} < \text{tol}$
    $\beta \leftarrow \frac{\delta_{\text{new}}}{\delta}$
    $\delta \leftarrow \delta_{\text{new}}$
    $d \leftarrow q + \beta d$                                                      ▷ Update search direction
end loop
```

6. Numerical results

In this chapter we demonstrate the efficiency and stability of our preconditioner on a number of test problems which are introduced in the following.

6.1. Test problems

As the first numerical test case, we define the following problem on the domain $\Omega = (0,1)^2 \subset \mathbb{R}^2$ and time interval $I = (0, 0.2)$. This problem was also considered in [4].

Problem 1 (P1). *Find u such that*

$$\begin{aligned} u_t - \Delta u &= f \quad \text{in} \quad I \times \Omega \\ u &= 0 \quad \text{on} \quad I \times \partial\Omega \end{aligned} \tag{6.1}$$

where

$$f := 2\sin(a\pi t)(x(1-x) + y(1-y)) + a\pi\cos(a\pi t)x(1-x)y(1-y) \tag{6.2}$$

and $a > 0$.

The analytical solution to this problem is

$$u = \sin(a\pi t)x(1-x)y(1-y). \tag{6.3}$$

In all following numerical examples, we fix the oscillation frequency of the solution to $a = 10$.

Remark 6.1. For fixed t, the solution and right-hand side of this problem can be fully resolved with fourth order elements. This means that the error exhibited by the fully discrete scheme will only be due to time discretization. We will use this fact to compute the experimental order of convergence in time.

Our second problem will be a 3d example to illustrate that our estimates hold in any space dimension. We consider the domain $\Omega = (0,1)^3$ and time interval $I = (0, 0.2)$ and pose the following problem:

Problem 2 (P2). *Find u such that*

$$\begin{aligned} u_t - \Delta u &= h \quad \text{in} \quad I \times \Omega \\ u &= 0 \quad \text{on} \quad I \times \partial\Omega \end{aligned} \tag{6.4}$$

where h is chosen such that the exact solution is of the form

$$u = \exp(t) \sin(\pi x) \sin(\pi y) \sin(\pi z). \tag{6.5}$$

The last problem will feature a convection part, *i.e.* the induced operator is no longer symmetric and positive definite. We use the same domain as in **P1**, namely $\Omega = (0,1)^2$ and time interval $I = (0, 0.2)$.

Problem 3 (P3). *For given $\varepsilon > 0$, find u such that*

$$\begin{aligned} u_t - \varepsilon\Delta u + b \cdot \nabla u &= g \quad \text{in} \quad I \times \Omega \\ u &= 0 \quad \text{on} \quad I \times \partial\Omega \end{aligned} \tag{6.6}$$

where $b = [y, -x]^\top$, and g is chosen such that the exact solution is given by

$$u = \sin(a\pi t) x(1-x) y(1-y) \tag{6.7}$$

as in problem **P1**.

6.2. Convergence results

To validate our implementation of the methods at hand, we computed their convergence rates, and compared them to known *a priori* estimates. We introduce the following notation.

For a numerical solution u_h, define the L^2 error $e_2(u_h) := \int_0^T ||u - u_h||^2_{L^2(\Omega)}$ and the discrete L^∞ error $e_\infty(u_h) := \max_n \|u(t_n) - u_h^n\|_{L^2(\Omega)}$.

Throughout this section, we use uniform time step size $\tau_n^{(k)} = \tau^{(k)}$ on each time refinement level k. For the numerical experiments, we choose $\tau^{(k)} = 1\text{e-}1/2^{k-1}$.

The experimental order of convergence (EOC) is given by the formula:

$$EOC_k^i := \frac{\log\left(\frac{e_i\left(u_h^{(k)}\right)}{e_i\left(u_h^{(k+1)}\right)}\right)}{\log\left(\frac{\tau^{(k)}}{\tau^{(k+1)}}\right)} \tag{6.8}$$

where $u_h^{(k)}$ is the discrete solution computed with time step size $\tau^{(k)}$.

We first consider problem **P1** discretized by polynomial elements of fourth order in space. This means that the discretization error is only due to time discretization and not space discretization.

The resulting convergence rates can be found in Tables 1 and 2. The numerical results exhibit second order convergence for the L^2-error for dG(1) and third order superconvergence at $t = t_n$ as expected. For cGP(2), third order convergence in the L^2-sense and fourth order superconvergence are also clearly visible.

TABLE 1. Experimental convergence rates for dG(1).

k	$e_2(u_h)$	EOC^2	$e_\infty(u_h)$	EOC^∞
1	1.43e-02	–	7.20e-03	–
2	3.40e-03	2.07	1.18e-03	2.61
3	8.73e-04	1.96	1.71e-04	2.78
4	2.21e-04	1.98	2.22e-05	2.95
5	5.53e-05	2.00	2.84e-06	2.96
6	1.38e-05	2.00	3.58e-07	2.99
7	3.46e-06	2.00	4.49e-08	3.00
8	8.64e-07	2.00	5.59e-09	3.01
9	2.16e-07	2.00	7.05e-10	2.99

TABLE 2. Experimental convergence rates for cGP(2).

k	$e_2(u_h)$	EOC^2	$e_\infty(u_h)$	EOC^∞
1	3.98e-03	–	4.07e-03	–
2	5.03e-04	2.98	1.98e-04	4.36
3	6.48e-05	2.96	1.44e-05	3.78
4	8.28e-06	2.97	8.84e-07	4.02
5	1.03e-06	3.00	5.58e-08	3.99
6	1.28e-07	3.01	3.49e-09	4.00
7	1.60e-08	3.00	2.39e-10	3.87

TABLE 3. Maximum number of operator evaluations per step needed to converge for dG(1), refinement level k (with fixed time step size).

Method\\k	1	2	3	4	5	6
BiCGStab (no)	32	60	118	210	436	886
BiCGStab (AMG)	6	8	8	12	10	8
CG (Schur)	6	36	161	668	2765	11 101
CG ($\mu = \mu_1$)	4	5	5	5	5	5
CG ($\mu = \mu_{\text{opt}}$)	4	4	4	4	4	4

6.3. Progression of residual norm

We consider the following solution strategies for (3.18), applied to problem **P1**:

- **BiCGStab (no)**: Unpreconditioned BiCGStab applied to the fully coupled system of equations (3.18),
- **BiCGStab (AMG)**: BiCGStab, applied to the fully coupled system of equations (3.18), but preconditioned by an algebraic multigrid method (see Sect. 5.2),
- **CG (Schur)**: Unpreconditioned CG, applied to the Schur complement formulation (3.19), and
- **CG** ($\mu \in \{\mu_1, \mu_{\text{opt}}\}$): CG, preconditioned using the left-right preconditioner A_μ as in Algorithm 1, where μ is chosen either to be the suboptimal value μ_1, or the optimal choice μ_{opt}.

In Table 3, we record the maximum number of operator evaluations per time step needed for each strategy on different refinement levels to reach a residual of norm < 1e-10. One operator evaluation includes the application of the Schur-complement operator and the inversion of the preconditioner, this happens once per iteration step for CG and twice for BiCGStab.

Let us mention that while for all numerical experiments in this subsection dG(1) was used, cGP(2) will lead to very similar results.

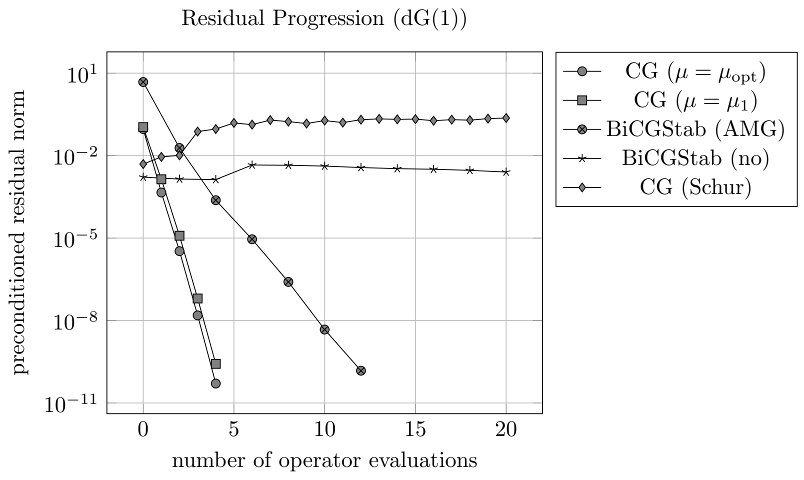

FIGURE 1. Progression of residual norm for different solution strategies for equation (3.18) for problem **P1**.

As can be seen, the iteration number depends on the mesh size for all unpreconditioned operators (BiCGStab (no), CG (Schur)). For the coupled system, which is an operator of second order in space, the number of iterations roughly doubles for each additional refinement step. For the fourth-order operator defined by the Schur-complement formulation, the number of iterations grows by a factor of four per refinement level.

For the AMG-preconditioned BiCGStab applied to the coupled system, we observe uniform behavior with respect to space discretization and typical iteration counts of 10.

When applying our proposed preconditioning scheme, we observe uniform behavior in all cases, however the number of Matrix-vector operations depends on the choice of μ and is lowest for the optimal parameter $\mu = \mu_{\text{opt}}$ as expected.

To differentiate further between the methods, we depict the residual progression on a typical refinement level for the aforementioned methods in Figure 1. As can be seen, the two unpreconditioned methods BiCGStab (no) and CG (Schur) do not significantly reduce the residual norm within the first 20 iterations. The AMG preconditioned BiCGStab converges to the desired tolerance within 12 operator evaluations, while the proposed preconditioned CG needs only 4 evaluations. The dependency on the parameter μ is not very pronounced in this case.

6.4. Iteration count for the preconditioner

In this section we are concerned with the sensitivity of the preconditioner with respect to space and time discretization parameters.

To this end, we first consider problem **P1**, discretized by Lagrange finite elements of order $p = 1, \dots, 4$ in space. We vary both mesh size and time step size, and trace the maximum number of iterations needed to reach a prescribed tolerance. In Table 4, the results are shown for dG(1), and in Table 5 for cGP(2). As expected, no dependency on polynomial degree, mesh or time step size occurs. 6 iterations suffice in every case to reach the desired tolerance. In the case of cGP(2), the lower condition number of the preconditioned system leads to a slightly faster convergence.

We repeated a similar numerical experiment for the 3-dimensional problem **P2**, with the same findings: as shown in Table 6, the number of iterations is never higher than 5 both for dG(1) and for cGP(2), confirming the uniform bound for the preconditioned system also in the 3d case.

To demonstrate the applicability of our method to more general operators, we proceed to apply our preconditioning scheme to **P3**. Due to the presence of convection, the preconditioned operator is no longer positive

TABLE 4. Maximum number of outer CG iterations for dG(1) with polynomial degree 1/2/3/4 for problem **P1**.

$\tau\backslash h$	2.83e-01	1.41e-01	7.07e-02	3.54e-02	1.77e-02	8.84e-03
1.00e-01	5/6/5/5	5/5/5/5	6/5/5/5	5/5/5/5	5/5/5/5	5/5/5/5
1.00e-02	4/5/6/6	5/6/6/6	5/6/6/6	6/6/6/6	6/6/6/6	6/6/6/6
1.00e-03	4/5/5/5	5/5/5/5	5/5/5/5	5/5/5/5	5/5/5/5	5/5/5/5
1.00e-04	3/3/4/4	3/4/4/4	3/4/4/4	4/4/4/4	4/4/4/4	4/4/4/4
1.00e-05	2/2/2/2	2/2/2/2	2/2/2/2	2/2/2/3	2/2/3/3	2/3/3/3
1.00e-06	1/2/2/2	2/2/2/2	2/2/2/2	2/2/2/2	2/2/2/2	2/2/2/2

TABLE 5. Maximum number of outer CG iterations for cGP(2) with polynomial degree 1/2/3/4 for problem **P1**.

$\tau\backslash h$	2.83e-01	1.41e-01	7.07e-02	3.54e-02	1.77e-02	8.84e-03
1.00e-01	5/5/5/5	5/5/5/5	5/5/5/5	5/5/5/5	5/5/5/5	5/5/5/5
1.00e-02	4/5/5/5	5/5/5/5	5/5/5/5	5/5/5/5	5/5/5/5	5/5/5/5
1.00e-03	4/5/5/5	4/5/5/5	5/5/5/5	5/5/5/5	5/5/5/5	5/5/5/5
1.00e-04	3/3/3/3	3/3/3/4	3/4/4/4	4/4/4/4	4/4/4/4	4/4/4/4
1.00e-05	2/2/2/2	2/2/2/2	2/2/2/2	2/2/2/2	2/2/2/2	2/2/2/2
1.00e-06	1/1/1/1	1/1/1/1	1/1/1/1	1/1/1/1	1/1/1/1	1/1/1/1

TABLE 6. Maximum number of outer CG iterations for dG(1)/cGP(2) for problem **P2** discretized by $\mathbb{V}_{h,1}$-functions.

$\tau\backslash h$	3.46e-01	1.73e-01	8.66e-02	4.33e-02
1.00e-01	5/5	5/5	5/4	5/4
1.00e-02	4/4	4/4	4/4	4/4
1.00e-03	4/4	4/4	4/4	4/4
1.00e-04	3/3	3/3	3/3	3/3
1.00e-05	2/2	2/2	2/2	2/2
1.00e-06	1/1	1/1	1/1	1/1

definite, and we cannot use CG in this case, but switch to BiCGStab, still using A_{opt} as a left-right preconditioner as before.

The influence of convection on the operator is varied by choosing different parameters $\varepsilon \in \{1,1\text{e-}2,1\text{e-}6\}$. The results are depicted in Table 7. Even for very small diffusion ($\varepsilon = 1\text{e-}6$) the maximum number of outer iterations is bounded by 5. Note that the number of iterations is typically somewhat lower than in the CG case, however BiCGStab needs two operator evaluations per iteration step whereas CG only needs one.

6.5. Computational costs

In this section we measure the computational costs of the preconditioned dG(1) method compared to the implicit Euler method. For this comparison, we solve problem **P1** discretized by $\mathbb{V}_{h,1}$-functions with both methods. As an inner solver for the preconditioned dG(1) method, we use CG preconditioned with AMG, whereas for the implicit Euler method the system is solved with the same implementation of CG preconditioned with AMG. For the execution time per time step both the application of the preconditioner and the Schur complement operator (which involves an inversion of the mass matrix) are included. Since the number of outer iterations is uniformly bounded, the computational overhead of the dG(1) method should only be a constant factor.

The computations were carried out on a workstation with an Intel Xeon E5 processor (2.7 GHz).

TABLE 7. Maximum number of BiCGStab iterations for dG(1) and cGP(2) with $\varepsilon = 1/1\text{e-}2/1\text{e-}6$ for problem **P3** (including convection) discretized by $\mathbb{V}_{h,1}$-functions.

		dG(1)		
$\tau\backslash h$	2.83e-01	1.41e-01	7.07e-02	3.54e-02
1.00e-01	3/4/5	3/4/5	3/4/5	3/4/5
1.00e-02	3/3/3	3/3/3	3/3/3	3/3/3
1.00e-03	2/2/2	2/2/2	3/2/2	2/2/2
1.00e-04	2/1/1	2/1/1	2/1/1	2/1/1
		cGP(2)		
1.00e-01	3/4/4	3/4/4	3/4/4	3/4/4
1.00e-02	3/2/2	3/2/2	3/3/3	3/3/3
1.00e-03	2/2/2	2/2/2	2/2/2	2/2/2
1.00e-04	1/1/1	2/1/1	2/1/1	2/1/1

TABLE 8. Computation time per time step of the implicit Euler and dG(1) method in comparison.

$\tau\backslash h$	1.41e-1	7.07e-2	3.54e-2	1.77e-2	8.84e-3
Avg. comp. time (ms) per time step for implicit Euler					
1.00e-01	2.21	4.67	10.49	26.63	102.19
1.00e-02	2.62	3.33	8.79	26.23	104.40
1.00e-03	2.01	3.01	7.26	24.02	97.72
1.00e-04	1.93	2.56	5.30	20.11	91.10
Avg. comp. time (ms) per time step for preconditioned dG(1)					
1.00e-01	11.43	29.27	61.33	211.47	815.21
1.00e-02	11.79	22.96	62.60	229.58	911.95
1.00e-03	8.91	16.02	48.26	180.72	729.92
1.00e-04	7.18	11.01	24.63	94.54	508.75
Ratio of comp. times of dG(1)/implicit Euler					
1.00e-01	5.18	6.26	5.85	7.94	7.98
1.00e-02	4.50	6.89	7.12	8.75	8.74
1.00e-03	4.44	5.33	6.65	7.52	7.47
1.00e-04	3.71	4.30	4.65	4.70	5.58

Table 8 summarizes our findings. As we have already demonstrated in the last section, the number of outer iterations for the uniformly preconditioned dG(1) method is bounded by approximately 5. In each outer iteration, two problems equivalent to one implicit Euler time step have to be solved. Thus, we expect the ratio of the computing times for both methods to be bounded by approximately 10. The numerical findings support this estimate.

To illustrate the benefits that the preconditioned higher order methods dG(1) and cGP(2) have over traditional low order methods (implicit Euler and Crank–Nicolson, respectively), we conducted the following benchmark: For each method, we measured the total CPU time (over all time steps) needed to reach a prescribed tolerance for the error measure e_∞ from Section 6.2 for problem **P1**. To this end, for each tolerance and method, we chose uniform time step sizes τ as large as possible to reach the prescribed error tolerance. For all methods, we solved the arising systems using the same solvers based on an AMG preconditioned CG. Our findings are depicted in Figure 2. We observed that for relatively large error tolerances $>10^{-2}$, lower order methods are competitive with the variational ones. This is to be expected, since computational costs for one time step are roughly 10 times larger for the variational methods, as was shown in the previous example. However, already for error tolerances in the range of 10^{-3}, variational methods start to outperform the traditional methods: for instance, to reach an error tolerance of 10^{-7}, the implicit Euler method needed 2132.5 s, Crank–Nicolson 4.65 s while

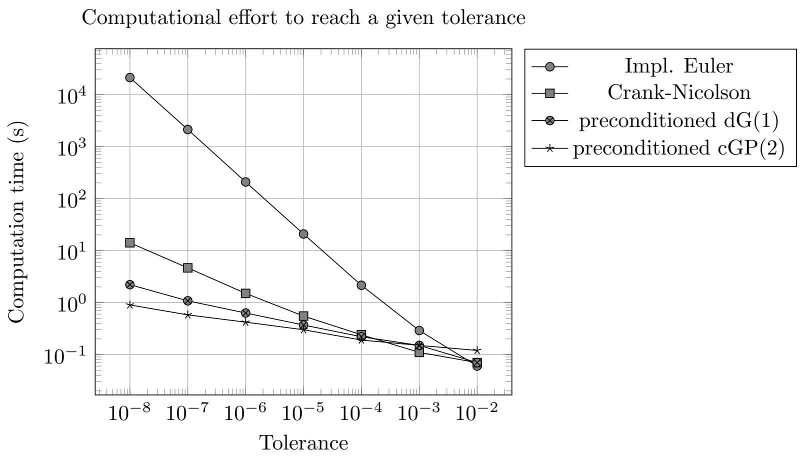

FIGURE 2. Progression of residual norm for different solution strategies for equation (3.18) for problem **P1**.

dG(1) needed 1.08 s and cGP(2) only 0.58 s. Consequently, even for moderate tolerances, the preconditioned higher-order methods considered in this paper are clearly superior over traditional lower order methods.

7. Conclusion

In this paper we have introduced an efficient preconditioning strategy for systems which arise from variational time discretization using first order discontinuous Galerkin (dG(1)) and second order continuous Galerkin Petrov (cGP(2)) methods. These time discretization schemes lead to a 2×2 coupled system of two unknowns.

The main ingredient of our method is a Schur-complement formulation for the *essential* unknown in the coupled system. The resulting ill-conditioned fourth-order operator is efficiently preconditioned by an inexact left-right factorization which avoids complex arithmetic and for which we can prove uniform bounds on the condition number.

Furthermore, the preconditioned operator retains symmetry and positive definiteness of the continuous operator. In contrast to other known methods this enables us to use the (symmetrically preconditioned) CG method as a solver.

The overall computational costs to solve the preconditioned operator are roughly equal to numerically inverting the operators resulting from an implicit Euler scheme twice.

In numerical experiments, we have demonstrated the efficiency and robustness of our method. While the analysis for the condition number is restricted to parabolic, time-independent operators, we successfully applied the method to a convection-diffusion problem.

References

[1] A. Aziz and P. Monk, Continuous finite elements in space and time for the heat equation. *Math. Comput.* **52** (1989) 255–274.

[2] E. Bänsch, P. Morin and R.H. Nochetto, Preconditioning a class of fourth order problems by operator splitting. *Numer. Math.* **118** (2011) 197–228.

[3] R.D. Falgout, J.E. Jones and U.M. Yang, The design and implementation of hypre, a library of parallel high performance preconditioners. In *Numerical solution of partial differential equations on parallel computers*. Springer (2006) 267–294.

[4] S. Hussain, F. Schieweck and S. Turek, Higher order galerkin time discretizations and fast multigrid solvers for the heat equation. *J. Numer. Math.* **19** (2011) 41–61.

[5] J.L. Lions and E. Magenes, Non-homogeneous Boundary Value Problems and Applications. Vol. I. Springer, New York (1972).

[6] A. Logg, K.-A. Mardal and G. Wells, Automated solution of differential equations by the finite element method: The fenics book. Vol. 84. Springer (2012).

[7] P. Lesaint and P.A. Raviart, On a Finite Element Method for Solving the Neutron Transport Equation. Analyse Numérique. University Paris VI, Labo (1974).

[8] T. Richter, A. Springer and B. Vexler, Efficient numerical realization of discontinuous galerkin methods for temporal discretization of parabolic problems. *Numer. Math.* (2012) 1–32.

[9] F. Schieweck, A-stable discontinuous Galerkin–Petrov time discretization of higher order. *J. Numer. Math.* **18** (2010) 25–57.

[10] F. Schieweck and G. Matthies, Higher order variational time discretizations for nonlinear systems of ordinary differential equations. Preprint 23/2011, Otto-von-Guericke-Universität Magdeburg (2011).

[11] D. Schötzau and C. Schwab, Time discretization of parabolic problems by the hp-version of the discontinuous galerkin finite element method. *SIAM J. Numer. Anal.* **38** (2000) 837–875.

[12] D. Schötzau and C. Schwab, hp-discontinuous galerkin time-stepping for parabolic problems. *C. R. Acad. Sci. Ser. I Math.* **333** (2001) 1121–1126.

[13] V. Thomée, Galerkin Finite Element Methods for Parabolic Problems. Number 1054 in *Springer Lect. Notes Math.* 2nd edition. Springer (1984).

[14] T. Werder, K. Gerdes, D. Schötzau and C. Schwab, hp-discontinuous galerkin time stepping for parabolic problems. *Comput. Methods Appl. Mech. Eng.* **190** (2001) 6685–6708.