Higher order Galerkin time discretizations and fast multigrid solvers for the heat equation

S. Hussain*, F. Schieweck†, S. Turek*

Abstract

We discuss numerical properties of continuous Galerkin-Petrov and discontinuous Galerkin time discretizations applied to the heat equation as a prototypical example for scalar parabolic partial differential equations. For the space discretization, we use biquadratic quadrilateral finite elements on general two-dimensional meshes. We discuss implementation aspects of the time discretization as well as efficient methods for solving the resulting block systems. Here, we compare a preconditioned BiCGStab solver as a Krylov space method with an adapted geometrical multigrid solver. Only the convergence of the multigrid method is almost independent of the mesh size and the time step leading to an efficient solution process. By means of numerical experiments we compare the different time discretizations with respect to accuracy and computational costs.

Keywords: Discontinuous Galerkin method, continuous Galerkin-Petrov method, heat equation, multigrid method

2000 Mathematics Subject Classification (MSC): 65M12, 65M55, 65M60

1 Introduction

A well-known approach to solve time dependent problems is the Galerkin method, see for instance the monograph [5]. In order to obtain a time marching process for this discretization, at least the test space needs to be discontinuous in time. In the discontinuous Galerkin method where solution and test space are the same, also the discrete solution space consists of discontinuous piecewise polynomials in time. Therefore, some jump terms appear in this discretization which however can be avoided if a continuous discrete solution space is combined with a discontinuous test space. Then the method is called a *Galerkin-Petrov method*.

*Institut für Angewandte Mathematik, Technische Universität Dortmund, Vogelpothsweg 87, 44227 Dortmund, Germany

†Institut für Analysis und Numerik, Otto-von-Guericke-Universität Magdeburg, Postfach 4120, D-39016 Magdeburg, Germany

In this paper, we compare this approach, which we call *continuous Galerkin-Petrov discretization* (cGP(k)-method, [4]), with the well-known *discontinuous Galerkin time discretization* (dG(k)-method, [5]). For the cGP(k)-method, the discrete solution space consists of continuous piecewise polynomial functions in time of degree $k \geq 1$ and the discrete test space of discontinuous polynomial functions of degree $k-1$. In the dG(k)-method, both the solution and test space are constructed by means of discontinuous polynomial functions of degree k. With respect to the computational costs, which mainly depend on the size of the resulting block system that has to be solved for each time interval, the cGP(k)-method is comparable to the dG(k-1)-method. However, concerning the discretization error in time, the accuracy of the cGP(k)-method is one order higher than that of the dG(k-1)-method. Furthermore, it is known that all cGP(k)-methods are A-stable, while all dG(k)-methods are even "strongly A-stable" (or L-stable), i.e., the dG-methods have better damping properties with respect to high frequency error components.

The approach of the cGP-method has already been used by Aziz and Monk [1] for the linear heat equation. They have proved optimal error estimates as well as superconvergence results, and the analysis of the dG-method can be found in [5]. In this paper, we present a numerical study of the higher order time discretizations cGP(1), cGP(2) and dG(1), taking into account aspects of numerical accuracy and efficiency of the corresponding solution methods for the resulting (coupled) linear systems. First of all, the cGP(1)-method is very close to the well-known Crank-Nicolson scheme: Both methods differ only in the choice of the unknown that is solved for on each time interval and in the way how the numerical integration of the right hand side is done. The cGP(1)-method is accurate of order 2 in the whole time interval as it is known for the Crank-Nicolson scheme. However, for the cGP(2)-method as well as in dG(1)-method, we have two unknowns on each time interval which have to be computed by solving a 2×2 block system. The cGP(2)-method is accurate of order 3 in the whole time interval and superconvergent of order 4 in the discrete time points [1]. The dG(1)-method is of order 2 in the whole time interval and superconvergent of order 3 in the discrete time points [5]. For all presented time stepping schemes, we use as spatial discretization the standard Galerkin Finite Element Method (FEM) with biquadratic quadrilateral elements.

To solve the associated linear block systems, we apply a preconditioned BiCGStab solver as standard Krylov space method and a geometrical (block) multigrid solver. The numerical experiments confirm that multigrid methods [2, 6] can be regarded as the most efficient iterative solvers since their rate of convergence is almost independent of the problem size which is characterized here by the mesh size of the space grid and the size of the time step.

2 The cGP-method for the heat equation

As a model problem we consider the heat equation: *Find* $u : \Omega \times [0,T] \to \mathbb{R}$ *such that*

$$\begin{aligned} d_t u - \triangle u &= f \quad \text{in} \quad \Omega \times (0,T), \\ u &= 0 \quad \text{on} \quad \partial\Omega \times [0,T], \\ u(x,0) &= u_0(x) \quad \text{for} \quad x \in \Omega, \end{aligned} \tag{1}$$

where $u(x,t)$ denotes the temperature in the point $x \in \Omega$ at time $t \in [0,T]$, $f : \Omega \times (0,T) \to \mathbb{R}$ a given source term and $u_0 : \Omega \to \mathbb{R}$ the initial temperature field at time $t = 0$. For simplicity, we assume homogeneous Dirichlet conditions at the boundary $\partial\Omega$ of a polygonal domain $\Omega \subset \mathbb{R}^2$.

We start with the time discretization of problem (1) which is of variational type. In the following, let $I = [0,T]$ be the time interval with some positive final time T. For a function $u : \Omega \times I \to \mathbb{R}$ and a fixed $t \in I$ we will denote by $u(t) := u(\cdot, t)$ the associated space function at time t which is an element of a suitable function space V. In case of the heat equation, this space is the Sobolev space $V = H_0^1(\Omega)$. In order to characterize functions $t \mapsto u(t)$ we define the space $C(I,V)$ as the space of continuous functions $u : I \to V$ equipped with the norm

$$\|u\|_{C(I,V)} := \sup_{t \in I} \|u(t)\|_V$$

and the space $L^2(I,V)$ containing discontinuous functions as

$$L^2(I,V) := \{u : I \to V : \ \|u\|_{L^2(I,V)} < \infty\}, \qquad \|u\|_{L^2(I,V)} := \left(\int_I \|u(t)\|_V^2 dt \right)^{1/2}.$$

In the time discretization, we decompose the time interval I into N subintervals $I_n := [t_{n-1}, t_n]$, where $n = 1, \ldots, N$ and $0 = t_0 < t_1 < \cdots < t_{N-1} < t_N = T$. The symbol τ will denote the *time discretization parameter* and will also be used as the maximum time step size $\tau := \max_{1 \le n \le N} \tau_n$, where $\tau_n := t_n - t_{n-1}$. Then, we approximate the solution $u : I \to V$ by means of a function $u_\tau : I \to V$ which is piecewise polynomial of some order k with respect to time, i.e., we are looking for u_τ in the discrete time space

$$X_\tau^k := \{u \in C(I,V) : \ u\big|_{I_n} \in \mathbb{P}_k(I_n, V) \quad \forall\, n = 1, \ldots, N\}, \tag{2}$$

where

$$\mathbb{P}_k(I_n, V) := \Big\{ u : I_n \to V \ : \ u(t) = \sum_{j=0}^{k} U^j t^j \,, \ \forall\, t \in I_n, \ U^j \in V, \ \forall\, j \Big\}.$$

We introduce the discrete time test space

$$Y_\tau^k := \{v \in L^2(I,V) : \ v\big|_{I_n} \in \mathbb{P}_{k-1}(I_n, V) \quad \forall\, n = 1, \ldots, N\} \tag{3}$$

consisting of piecewise polynomials of order $k-1$ which are globally discontinuous at the end points of the time intervals. Now, we multiply the first equation in (1) with a test function $v_\tau \in Y_\tau^k$, integrate over $\Omega \times I$, use Fubini's Theorem and partial space integration of the Laplacian term and obtain the following *time discrete problem*: *Find* $u_\tau \in X_\tau^k$ *such that* $u_\tau(0) = u_0$ *and*

$$\int_0^T \Big\{ (d_t u_\tau(t), v_\tau(t))_\Omega + a(u_\tau(t), v_\tau(t)) \Big\} dt = \int_0^T (f(t), v_\tau(t))_\Omega \, dt \qquad \forall\, v_\tau \in Y_\tau^k, \tag{4}$$

where $(\cdot,\cdot)_\Omega$ denotes the usual inner product in $L^2(\Omega)$ and $a(\cdot,\cdot)$ the bilinear form on $V \times V$ defined as

$$a(u,v) := \int_\Omega \nabla u \cdot \nabla v \, dx \qquad \forall\, u, v \in V.$$

We will call this discretization the *exact continuous Galerkin-Petrov method* of *order* k or briefly the "*exact* cGP(k)-*method*". The name Galerkin-Petrov is due to the fact that the test space Y_τ^k is different from the ansatz space X_τ^k. With "exact" we indicate that the time integral at the right hand side is evaluated exactly.

Since the discrete test space Y_τ^k is discontinuous, problem (4) can be solved in a time marching process where successively local problems on the time intervals are solved. Therefore, we choose test functions $v_\tau(t) = v\psi_{n,i}(t)$ with an arbitrary time independent $v \in V$ and a scalar function $\psi_{n,i} : I \to \mathbb{R}$ which is zero on $I \setminus I_n$ and a polynomial of order less or equal $k-1$ on I_n. Then, we obtain from (4) the "I_n-problem": *Find* $u_\tau|_{I_n} \in \mathbb{P}_k(I_n, V)$ *such that*

$$\int_{I_n} \Big\{ (d_t u_\tau(t), v)_\Omega + a(u_\tau(t), v) \Big\} \psi_{n,i}(t) \, dt = \int_{I_n} (f(t), v)_\Omega \, \psi_{n,i}(t) \, dt \qquad \forall\, v \in V \tag{5}$$

for $i = 1, \ldots, k$, with the "initial condition" $u_\tau|_{I_n}(t_{n-1}) = u_\tau|_{I_{n-1}}(t_{n-1})$ for $n \geq 2$ or $u_\tau|_{I_n}(t_{n-1}) = u_0$ for $n = 1$.

To determine $u_\tau|_{I_n}$ we represent it by a polynomial ansatz

$$u_\tau(t) := \sum_{j=0}^{k} U_n^j \phi_{n,j}(t) \qquad \forall\, t \in I_n, \tag{6}$$

where the "coefficients" U_n^j are elements of the Hilbert space V and the real functions $\phi_{n,j} \in \mathbb{P}_k(I_n)$ are the Lagrange basis functions with respect to $k+1$ suitable nodal points $t_{n,j} \in I_n$ satisfying the conditions

$$\phi_{n,j}(t_{n,i}) = \delta_{i,j}, \qquad i,j = 0, \ldots, k \tag{7}$$

with the Kronecker symbol $\delta_{i,j}$. In [4], the $t_{n,j}$ have been chosen as the quadrature points of the $(k+1)$-point Gauß-Lobatto formula on I_n. Here, we take another choice: For an easy treatment of the initial condition for (5), we set $t_{n,0} = t_{n-1}$. Then, the initial condition is equivalent to the condition

$$U_n^0 = u_\tau|_{I_{n-1}}(t_{n-1}) \quad \text{if} \quad n \geq 2 \qquad \text{or} \qquad U_n^0 = u_0 \quad \text{if} \quad n = 1. \tag{8}$$

The other points $t_{n,1}, \ldots, t_{n,k}$ are chosen as the quadrature points of the k-point Gauß formula on I_n. This formula is exact if the function to be integrated is a polynomial of degree less or equal $2k-1$. From the representation (6) we get

$$\int_{I_n} (d_t u_\tau(t), v)_\Omega \, \psi_{n,i}(t) dt = \sum_{j=0}^{k} \left(U_n^j, v\right)_\Omega \int_{I_n} \phi'_{n,j}(t)\psi_{n,i}(t)\, dt \qquad \forall\, v \in V. \tag{9}$$

We define the basis functions $\phi_{n,j} \in \mathbb{P}_k(I_n)$ of (6) via the affine reference transformation $T_n : \hat{I} \to I_n$ where $\hat{I} := [-1, 1]$ and

$$t = T_n(\hat{t}) := \frac{t_{n-1} + t_n}{2} + \frac{\tau_n}{2}\hat{t} \in I_n \qquad \forall\, \hat{t} \in \hat{I},\ n = 1, \ldots, N. \tag{10}$$

Let $\hat{\phi}_j \in \mathbb{P}_k(\hat{I})$, $j = 0, \ldots, k$, denote the basis functions satisfying the conditions

$$\hat{\phi}_j(\hat{t}_i) = \delta_{i,j}, \qquad i, j = 0, \ldots, k, \tag{11}$$

where $\hat{t}_0 = -1$ and $\hat{t}_i$, $i = 1, \ldots, k$, are the standard *Gauß* quadrature points for the reference interval $\hat{I}$. Then, we define the basis functions on the original time interval I_n by

$$\phi_{n,j}(t) := \hat{\phi}_j(\hat{t}) \qquad \text{with} \qquad \hat{t} := T_n^{-1}(t) = \frac{2}{\tau_n}\left(t - \frac{t_n - t_{n-1}}{2}\right) \in \hat{I}. \tag{12}$$

Similarly, we define the test basis functions $\psi_{n,i}$ by suitable reference basis functions $\hat{\psi}_i \in \mathbb{P}_{k-1}(\hat{I})$, i.e.,

$$\psi_{n,i}(t) := \hat{\psi}_i(T_n^{-1}(t)) \qquad \forall\, t \in I_n,\ i = 1, \ldots, k. \tag{13}$$

For practical computations, we have to approximate the right hand side in the exact cGP(k)-method (5) by some numerical integration. To this end, we replace the function $f(t)$ by the time-polynomial $\pi_k f \in \mathbb{P}_k(I_n, L^2(\Omega))$ defined as the Lagrange interpolate

$$\pi_k f(t) := \sum_{j=0}^{k} f(t_{n,j})\phi_{n,j}(t) \qquad \forall\, t \in I_n.$$

Now, we transform all integrals in (5) to the reference interval $\hat{I}$ and obtain the following system of equations for the "coefficients" $U_n^j \in V$ in the ansatz (6)

$$\sum_{j=0}^{k} \left\{ \alpha_{i,j} \left(U_n^j, v\right)_\Omega + \frac{\tau_n}{2}\beta_{i,j} a(U_n^j, v) \right\} = \frac{\tau_n}{2} \sum_{j=0}^{k} \beta_{i,j} \left(f(t_{n,j}), v\right)_\Omega \qquad \forall\, v \in V \tag{14}$$

where $i = 1, \ldots, k$,

$$\alpha_{i,j} := \int_{\hat{I}} \hat{\phi}'_j(\hat{t})\hat{\psi}_i(\hat{t})\, d\hat{t}, \qquad \beta_{i,j} := \int_{\hat{I}} \hat{\phi}_j(\hat{t})\hat{\psi}_i(\hat{t})\, d\hat{t} \tag{15}$$

and the "coefficient" $U_n^0 \in V$ is known. Due to the polynomial degree of $\hat{\phi}_j$ and $\hat{\psi}_i$ we can compute the integrals for $\alpha_{i,j}$ and $\beta_{i,j}$ exactly by means of the k-point Gauß formula with

weights $\hat{w}_\mu$ and points $\hat{t}_\mu$, $\mu = 1, \ldots, k$. If we choose the test functions $\hat{\psi}_i \in \mathbb{P}_{k-1}(\hat{I})$ in (15) such that

$$\hat{\psi}_i(\hat{t}_\mu) = (\hat{w}_i)^{-1}\delta_{i,\mu} \qquad \forall\, i, \mu \in \{1, \ldots, k\},$$

we get from (15) that

$$\alpha_{i,j} = \hat{\phi}_j'(\hat{t}_i), \qquad \beta_{i,j} = \delta_{i,j}, \qquad 1 \le i \le k,\ 0 \le j \le k.$$

Then, the system (14) is equivalent to the following coupled system of equations for the k unknown "coefficients" $U_n^j \in V$, $j = 1, \ldots, k$,

$$\sum_{j=0}^{k} \alpha_{i,j} \left(U_n^j, v\right)_\Omega + \frac{\tau_n}{2} a(U_n^i, v) = \frac{\tau_n}{2} \left(f(t_{n,i}), v\right)_\Omega \qquad \forall\, v \in V, \quad i = 1, \ldots, k, \tag{16}$$

where $U_n^0 = u_\tau(t_{n-1})$ for $n > 1$ and $U_1^0 = u_0$. In the following, we specify the cGP(k)-method for the cases $k = 1$ and $k = 2$.

2.1 cGP(1)-method

We use the one-point Gauß quadrature formula with the point $\hat{t}_1 = 0$ and $t_{n,1} = t_{n-1} + \frac{\tau_n}{2}$. Then, we get $\alpha_{1,0} = -1$ and $\alpha_{1,1} = 1$. Thus, equation (16) leads to the following equation for the one "unknown" $U_n^1 = u_\tau(t_{n-1} + \frac{\tau_n}{2}) \in V$

$$\left(U_n^1, v\right)_\Omega + \frac{\tau_n}{2} a(U_n^1, v) = \frac{\tau_n}{2} \left(f(t_{n,1}), v\right)_\Omega + \left(U_n^0, v\right)_\Omega \qquad \forall\, v \in V. \tag{17}$$

Once we have determined the solution U_n^1, we get the solution at discrete time t_n by means of linear extrapolation

$$u_\tau(t_n) = 2U_n^1 - U_n^0, \tag{18}$$

where U_n^0 is the initial value at the time interval $[t_{n-1}, t_n]$ coming from the previous time interval or the initial value u_0. If we would replace $f(t_{n,1})$ by the mean value $(f(t_{n-1}) + f(t_n))/2$, which means that we replace the one-point Gauß quadrature of the right hand side by the Trapezoidal rule, the resulting cGP(1)-method would be equivalent to the well-known *Crank-Nicolson scheme*.

2.2 cGP(2)-method

We use the 2-point Gauß quadrature formula with the points $\hat{t}_1 = -\frac{1}{\sqrt{3}}$ and $\hat{t}_2 = \frac{1}{\sqrt{3}}$. Then, we obtain the coefficients

$$(\alpha_{i,j}) = \begin{pmatrix} -\sqrt{3} & \frac{3}{2} & \frac{2\sqrt{3}-3}{2} \\ \sqrt{3} & \frac{-2\sqrt{3}-3}{2} & \frac{3}{2} \end{pmatrix} \qquad i = 1, 2,\ j = 0, 1, 2.$$

On the time interval $I_n = [t_{n-1}, t_n]$ we have to solve for the two "unknowns" $U_n^j = u_\tau(t_{n,j})$ with $t_{n,j} := T_n(\hat{t}_j)$ for $j = 1, 2$. The coupled system for $U_n^1, U_n^2 \in V$ reads

$$\begin{aligned} \left\{\alpha_{1,1}\left(U_n^1, v\right)_\Omega + \tfrac{\tau_n}{2} a(U_n^1, v)\right\} + \alpha_{1,2}\left(U_n^2, v\right)_\Omega &= \tfrac{\tau_n}{2}\left(f(t_{n,1}), v\right)_\Omega - \alpha_{1,0}\left(U_n^0, v\right)_\Omega, \\ \alpha_{2,1}\left(U_n^1, v\right)_\Omega + \left\{\alpha_{2,2}\left(U_n^2, v\right)_\Omega + \tfrac{\tau_n}{2} a(U_n^2, v)\right\} &= \tfrac{\tau_n}{2}\left(f(t_{n,2}), v\right)_\Omega - \alpha_{2,0}\left(U_n^0, v\right)_\Omega, \end{aligned} \tag{19}$$

which has to be satisfied for all $v \in V$. Once we have determined the solution (U_n^1, U_n^2), we get the solution at discrete time t_n by means of quadratic extrapolation

$$u_\tau(t_n) = U_n^0 + \sqrt{3}(U_n^2 - U_n^1), \tag{20}$$

where U_n^0 is the initial value at the time interval I_n coming from the previous time interval or the initial value u_0.

3 Discontinuous Galerkin methods

In this section we describe the details of the discontinuous Galerkin method dG(k-1) which is with respect to the size of the coupled system comparable to the cGP(k)-method. Here the discrete solution space is the same as the test space Y_τ^k of the cGP(k)-method defined in (3). Therefore, the discrete solution u_τ is on each time interval I_n a polynomial in time of degree $k-1$ with a representation

$$u_\tau(t) := \sum_{j=1}^{k} U_n^j \phi_{n,j}(t) \qquad \forall\, t \in I_n, \tag{21}$$

where the "coefficients" U_n^j are elements of the Hilbert space V and the real functions $\phi_{n,j} \in \mathbb{P}_{k-1}(I_n)$ are the Lagrange basis functions with respect to k suitable nodal points $t_{n,j} \in I_n$ satisfying the conditions $\phi_{n,j}(t_{n,i}) = \delta_{i,j}$ for all $i, j = 1, \ldots, k$. Since u_τ is discontinuous at t_n, we define the following left and right sided values u_n^- and u_n^+ and the jump $[u_\tau]_n$ as:

$$u_n^- := \lim_{t \to t_n - 0} u_\tau(t), \qquad u_n^+ := \lim_{t \to t_n + 0} u_\tau(t), \qquad [u_\tau]_n := u_n^+ - u_n^-.$$

Then, the discontinuous Galerkin method dG(k-1) reads: *Find* $u_\tau \in Y_\tau^k$ *such that* $u_\tau(0) = u_0$ *and for all* $v_\tau \in Y_\tau^k$ *it holds*

$$\sum_n \int_{I_n} \Big\{ \left(d_t u_\tau(t), v_\tau(t)\right)_\Omega + a(u_\tau(t), v_\tau(t)) \Big\} dt + \sum_n \left([u_\tau]_{n-1}, v_{n-1}^+\right)_\Omega = \int_0^T \left(f(t), v_\tau(t)\right)_\Omega dt.$$

To decouple this formulation we choose test functions $v_\tau(t) = v\psi_{n,i}(t)$ with an arbitrary t-independent $v \in V$ and a scalar piecewise polynomial function $\psi_{n,i}$ which is zero on $I \setminus I_n$ and a basis function of the polynomial space $\mathbb{P}_{k-1}$ on I_n. Then, using the fact that $U_n^0 := u_{n-1}^-$

is known from the previous time interval or the initial value u_0, the solution of the dG(k-1)-method can be determined by successively solving a local problem on each time interval I_n. This "I_n-problem" reads: *Find* $u_\tau \in \mathbb{P}_{k-1}(I_n, V)$ *such that for all* $v \in V$ *and all* $i = 1, \ldots, k$ *holds*

$$\begin{aligned}\int_{I_n} \Big\{ (d_t u_\tau(t), v)_\Omega + a(u_\tau(t), v) \Big\} \psi_{n,i}(t)\, dt + \big(u_{n-1}^+, v\big)_\Omega \psi_{n,i}(t_{n-1}) \\ = \big(U_n^0, v\big)_\Omega \psi_{n,i}(t_{n-1}) + \int_{I_n} (f(t), v)_\Omega \psi_{n,i}(t)\, dt.\end{aligned} \tag{22}$$

As in the cGP-method we define the basis functions $\phi_{n,j} \in \mathbb{P}_{k-1}(I_n)$ in (21) by means of reference basis functions $\hat{\phi}_j \in \mathbb{P}_{k-1}(\hat{I})$ via the transformation $T_n : \hat{I} \to I_n$ given in (10). The $\hat{\phi}_j$, $j = 1, \ldots, k$, are chosen as the Lagrange basis functions associated with the integration points $\hat{t}_\mu$ of the k-point Gauß quadrature rule on $\hat{I} = [-1, 1]$, i.e., they satisfy the conditions $\hat{\phi}_j(\hat{t}_\mu) = \delta_{j,\mu}$ for all $j, \mu = 1, \ldots, k$. Similarly, we define the test basis functions $\psi_{n,i}$ by means of reference basis functions $\hat{\psi}_i \in \mathbb{P}_{k-1}(\hat{I})$. We use the choice $\hat{\psi}_i := (\hat{w}_i)^{-1} \hat{\phi}_i$ where the $\hat{w}_i$ denote the weights of the the k-point Gauß formula on $\hat{I}$.

Now, if we insert the representation (21) of u_τ into the I_n-problem (22) and transform the integrals to the reference interval $\hat{I}$, we obtain the following system of equations for the "coefficients" $U_n^j \in V$, $j = 1, \ldots, k$

$$\sum_{j=1}^{k} \Big\{ \alpha_{i,j} \left(U_n^j, v\right)_\Omega + \frac{\tau_n}{2} \beta_{i,j} a(U_n^j, v) + c_j d_i \left(U_n^j, v\right)_\Omega \Big\} = d_i \left(U_n^0, v\right)_\Omega + \frac{\tau_n}{2} \int_{\hat{I}} \Big(\hat{f}, v\Big)_\Omega \hat{\psi}_i d\hat{t},$$

where $i = 1, \ldots, k$ and

$$\alpha_{i,j} := \int_{\hat{I}} \hat{\phi}_j'(\hat{t}) \hat{\psi}_i(\hat{t})\, d\hat{t}, \qquad \beta_{i,j} := \int_{\hat{I}} \hat{\phi}_j(\hat{t}) \hat{\psi}_i(\hat{t})\, d\hat{t}, \qquad c_j := \hat{\phi}_j(-1), \qquad d_i := \hat{\psi}_i(-1).$$

We approximate the integral on the right hand side by the k-point Gauß formula and get due to the special choice of $\hat{\psi}_i$

$$\int_{\hat{I}} \Big(\hat{f}(\hat{t}), v\Big)_\Omega \hat{\psi}_i(\hat{t})\, d\hat{t} \;\approx\; \sum_{\mu=1}^{k} \hat{w}_\mu \left(f(t_{n,\mu}), v\right)_\Omega \hat{\psi}_i(\hat{t}_\mu) = \left(f(t_{n,i}), v\right)_\Omega, \tag{23}$$

where the $t_{n,\mu} := T_n(\hat{t}_\mu) \in I_n$ denote the mapped Gauß points. Since the k-point Gauß quadrature formula is exact for the integrals defining $\alpha_{i,j}$ and $\beta_{i,j}$, we conclude from the Kronecker delta properties of $\hat{\phi}_j$ and $\hat{\psi}_i$ that

$$\alpha_{i,j} = \hat{\phi}_j'(\hat{t}_i), \qquad \beta_{i,j} = \delta_{i,j}, \qquad c_j = \hat{\phi}_j(-1), \qquad d_i = \frac{1}{\hat{w}_i} c_i \tag{24}$$

for $i, j = 1, \ldots, k$.

Summarizing all pieces, we get the following version of the dG(k-1)-method with numerically integrated right hand side on the time interval $I_n = [t_{n-1}, t_n]$. For a given value

$U_n^0 := u_{n-1}^-$ known from the previous time interval or the initial value u_0, find k unknowns $U_n^1, \ldots, U_n^k \in V$ such that for all $v \in V$ and all $i = 1, \ldots, k$, holds

$$\sum_{j=1}^{k} \left(\alpha_{i,j} + c_j d_i\right) \left(U_n^j, v\right)_\Omega + \frac{\tau_n}{2} a(U_n^i, v) = d_i \left(U_n^0, v\right)_\Omega + \frac{\tau_n}{2} \left(f(t_{n,i}), v\right)_\Omega . \tag{25}$$

Once we have solved this system, we can compute by means of the ansatz (21) the left side value u_n^- of u_τ at time t_n. Then, we enter the next time interval $I_{n+1} = [t_n, t_{n+1}]$ and set the initial value to $U_{n+1}^0 := u_n^-$.

Finally, we will specify the case $k = 2$ leading to the dG(1)-method which we will use in our numerical experiments. Here, the discrete solution u_τ is piecewise linear, i.e., we have to compute two "coefficients" on each time interval to define the linear approximation. The associated 2-point Gauß quadrature formula has the weights $\hat{w}_1 = \hat{w}_2 = 1$ and integration points $\hat{t}_1 = -\frac{1}{\sqrt{3}}$, $\hat{t}_2 = \frac{1}{\sqrt{3}}$. With the abbreviation $\gamma_{i,j} := \alpha_{i,j} + c_j d_i$ we obtain for the coefficients in (25) the values

$$(\alpha_{i,j}) = \begin{pmatrix} \frac{-\sqrt{3}}{2} & \frac{\sqrt{3}}{2} \\ \frac{-\sqrt{3}}{2} & \frac{\sqrt{3}}{2} \end{pmatrix}, \quad (c_i) = (d_i) = \begin{pmatrix} \frac{\sqrt{3}+1}{2} \\ \frac{-\sqrt{3}+1}{2} \end{pmatrix}, \quad (\gamma_{i,j}) = \begin{pmatrix} 1 & \frac{\sqrt{3}-1}{2} \\ \frac{-\sqrt{3}-1}{2} & 1 \end{pmatrix}.$$

Thus, in the **dG(1)-method**, one has to determine on the time interval I_n the two "unknowns" $U_n^1, U_n^2 \in V$ as the solution of the following coupled system

$$\begin{aligned} \left\{\gamma_{1,1} \left(U_n^1, v\right)_\Omega + \tfrac{\tau_n}{2} a(U_n^1, v)\right\} + \gamma_{1,2} \left(U_n^2, v\right)_\Omega &= d_1 \left(U_n^0, v\right)_\Omega + \tfrac{\tau_n}{2} \left(f(t_{n,1}), v\right)_\Omega , \\ \gamma_{2,1} \left(U_n^1, v\right)_\Omega + \left\{\gamma_{2,2} \left(U_n^2, v\right)_\Omega + \tfrac{\tau_n}{2} a(U_n^2, v)\right\} &= d_2 \left(U_n^0, v\right)_\Omega + \tfrac{\tau_n}{2} \left(f(t_{n,2}), v\right)_\Omega , \end{aligned} \tag{26}$$

which has to be satisfied for all $v \in V$. Once we have solved the above system, we get the solution at the time t_n by means of the following linear extrapolation with the "values" $U_n^j = u_\tau(t_{n,j})$, $j = 1, 2$

$$u_\tau(t_n) = \frac{\sqrt{3}+1}{2} U_n^2 - \frac{\sqrt{3}-1}{2} U_n^1. \tag{27}$$

4 Space Discretization by FEM

After discretizing equation (1) in time, we now apply the finite element method to discretize each of the "I_n-problem" in space. To this end, let $V_h \subset V$ denote a finite element space. In the numerical experiments, V_h will be defined by biquadratic finite elements on a quadrilateral mesh T_h for the computational domain Ω. Each "I_n-problem" for the cGP(k)-method or the dG(k-1)-method has the structure: *For given $U_n^0 \in V$, find $U_n^1, \ldots, U_n^k \in V$ such that*

$$\sum_{j=1}^{k} \gamma_{i,j} \left(U_n^j, v\right)_\Omega + \frac{\tau_n}{2} a(U_n^i, v) = d_i \left(U_n^0, v\right)_\Omega + \frac{\tau_n}{2} \left(f(t_{n,i}), v\right)_\Omega \qquad \forall\, v \in V, \tag{28}$$

for all $i = 1, \ldots, k$, where $\gamma_{i,j}$ and d_i are given constants. In the space discretization, each $U_n^j \in V$ is approximated by a finite element function $U_{n,h}^j \in V_h$ and the fully discrete "I_n-problem" reads: *For given $U_{n,h}^0 \in V_h$, find $U_{n,h}^1, \ldots, U_{n,h}^k \in V_h$ such that*

$$\sum_{j=1}^{k} \gamma_{i,j} \left(U_{n,h}^j, v_h \right)_\Omega + \frac{\tau_n}{2} a(U_{n,h}^i, v_h) = d_i \left(U_{n,h}^0, v_h \right)_\Omega + \frac{\tau_n}{2} \left(f(t_{n,i}), v_h \right)_\Omega \qquad \forall\, v_h \in V_h, \quad (29)$$

for all $i = 1, \ldots, k$. Once we have solved this system, we can compute for each time $t \in I_n$ a finite element approximation $u_{\tau,h}(t) \in V_h$ of the time discrete solution $u_\tau(t) \in V$. To this end, we replace the "constants" $U_n^j \in V$ in the ansatz of $u_\tau(t)$ by the space discrete "constants" $U_{n,h}^j \in V_h$.

In the following, we will write the problem (29) as a linear algebraic block system. Let $b_\mu \in V_h$, $\mu = 1, \ldots, m_h$, denote the finite element basis functions and $\underline{U}_n^j \in \mathbb{R}^{m_h}$ the nodal vector of $U_{n,h}^j \in V_h$ such that

$$U_{n,h}^j(x) = \sum_{\mu=1}^{m_h} (\underline{U}_n^j)_\mu b_\mu(x) \qquad \forall\, x \in \Omega.$$

Furthermore, let us introduce the mass matrix $M \in \mathbb{R}^{m_h \times m_h}$, the discrete Laplacian matrix $L \in \mathbb{R}^{m_h \times m_h}$ and the vector $F_n^i \in \mathbb{R}^{m_h}$ as

$$M_{\nu,\mu} := (b_\mu, b_\nu)_\Omega, \qquad L_{\nu,\mu} := a(b_\mu, b_\nu), \qquad (F_n^i)_\nu := (f(t_{n,i}), b_\nu)_\Omega. \quad (30)$$

Then the fully discrete "I_n-problem" is equivalent to the following linear $k \times k$ block system: *For given $\underline{U}_n^0 \in \mathbb{R}^{m_h}$, find $\underline{U}_n^1, \ldots, \underline{U}_n^k \in \mathbb{R}^{m_h}$ such that*

$$\sum_{j=1}^{k} \gamma_{i,j} M \underline{U}_n^j + \frac{\tau_n}{2} L \underline{U}_n^i = d_i M \underline{U}_n^0 + \frac{\tau_n}{2} F_n^i, \qquad \forall\, i = 1, \ldots, k. \quad (31)$$

The vector $\underline{U}_n^0$ is defined as the finite element nodal vector of the fully discrete solution $u_{\tau,h}(t_{n-1})$ computed from the previous time interval $[t_{n-2}, t_{n-1}]$ if $n \geq 2$ or from a finite element interpolation of the initial data u_0 if $n = 1$.

In the following, we will present the resulting block systems for the cGP(1), cGP(2) and dG(1) method which are used in our numerical experiments.

4.1 cGP(1)-method

The problem on time interval I_n reads: *For given $\underline{U}_n^0 \in \mathbb{R}^{m_h}$, find $\underline{U}_n^1 \in \mathbb{R}^{m_h}$ such that*

$$\left(M + \frac{\tau_n}{2} L \right) \underline{U}_n^1 = \frac{\tau_n}{2} F_n^1 + M \underline{U}_n^0. \quad (32)$$

Once we have determined the solution $\underline{U}_n^1$, we compute the nodal vector $\underline{U}_{n+1}^0$ of the fully discrete solution $u_{\tau,h}$ at the time t_n by using the following linear extrapolation

$$u_{\tau,h}(t_n) \sim \underline{U}_{n+1}^0 = 2\underline{U}_n^1 - \underline{U}_n^0.$$

4.2 cGP(2)-method

The 2×2 block system on time interval I_n reads: *For given* $\underline{U}_n^0 \in \mathbb{R}^{m_h}$, *find* $\underline{U}_n^1, \underline{U}_n^2 \in \mathbb{R}^{m_h}$ *such that*

$$\begin{pmatrix} 3M + \tau_n L & \left(2\sqrt{3} - 3\right) M \\ \left(-2\sqrt{3} - 3\right) M & 3M + \tau_n L \end{pmatrix} \begin{pmatrix} \underline{U}_n^1 \\ \underline{U}_n^2 \end{pmatrix} = \begin{pmatrix} R_n^1 \\ R_n^2 \end{pmatrix} \tag{33}$$

where

$$\begin{aligned} R_n^1 &= \tau_n F_n^1 + 2\sqrt{3} M \underline{U}_n^0 \\ R_n^2 &= \tau_n F_n^2 - 2\sqrt{3} M \underline{U}_n^0. \end{aligned}$$

Once we have determined the solution $(\underline{U}_n^1, \underline{U}_n^2)$, we compute the nodal vector $\underline{U}_{n+1}^0$ of the fully discrete solution $u_{\tau,h}$ at the time t_n by using the following quadratic extrapolation

$$u_{\tau,h}(t_n) \sim \underline{U}_{n+1}^0 = \underline{U}_n^0 + \sqrt{3}(\underline{U}_n^2 - \underline{U}_n^1).$$

4.3 dG(1)-method

The 2×2 block system on time interval I_n reads: *For given* $\underline{U}_n^0 \in \mathbb{R}^{m_h}$, *find* $\underline{U}_n^1, \underline{U}_n^2 \in \mathbb{R}^{m_h}$ *such that*

$$\begin{pmatrix} 2M + \tau_n L & \left(\sqrt{3} - 1\right) M \\ \left(-\sqrt{3} - 1\right) M & 2M + \tau_n L \end{pmatrix} \begin{pmatrix} \underline{U}_n^1 \\ \underline{U}_n^2 \end{pmatrix} = \begin{pmatrix} R_n^1 \\ R_n^2 \end{pmatrix} \tag{34}$$

where

$$\begin{aligned} R_n^1 &= \tau_n F_n^1 + \left(\sqrt{3} + 1\right) M \underline{U}_n^0 \\ R_n^2 &= \tau_n F_n^2 + \left(-\sqrt{3} + 1\right) M \underline{U}_n^0. \end{aligned}$$

Once we have determined the solution $(\underline{U}_n^1, \underline{U}_n^2)$, we compute the nodal vector $\underline{U}_{n+1}^0$ of the left side limit of the fully discrete solution $u_{\tau,h}$ at the time t_n by using the following linear extrapolation

$$u_{\tau,h}^-(t_n) \sim \underline{U}_{n+1}^0 = \frac{\sqrt{3}+1}{2} U_n^2 - \frac{\sqrt{3}-1}{2} U_n^1.$$

5 Solution of the linear systems

The resulting linear systems in each time interval $[t_{n-1}, t_n]$, which are 2x2 block matrices in the case of the cGP(2) and dG(1) approach, are treated either by preconditioned Krylov-space or geometrical multigrid solvers. Among the Krylov-space methods we employ the classical BiCGStab solver (which is chosen in view of future nonsymmetric matrices due to convection-diffusion or even the Navier-Stokes equations) with the SSOR method as preconditioner. However, such 'single grid' methods cannot treat the associated systems in an optimally efficient manner since the corresponding condition numbers depend also on the mesh size so that it is expected to become very expensive for higher mesh levels.

To overcome this difficulty, we utilize a geometrical multigrid solver with corresponding (block) smoothers and grid transfer operators. Multigrid methods are regarded as the most efficient iterative methods for the solution of large linear systems arising from the discretization of partial differential equations, particularly of elliptic type. In this paper, we use the standard refinement scheme (see [6]) for the grid hierarchies, and for the smoothing operator, we choose the same SSOR method as for the preconditioned Krylov-space solver; however, also corresponding standard block variants of Jacobi, SOR and ILU methods can be easily applied. Moreover, we use the canonical grid transfer routines regarding the chosen FEM space which treat both solution components separately in the case of the cGP(2) and dG(1) approaches (see [3] for the details, particularly regarding the grid transfer for the biquadratic finite elements). Finally, the coarse grid problem is solved by a direct solver.

6 Numerical results

In this section, we perform several numerical tests in order to compare the accuracy of our time discretization schemes. As the first test example we consider problem (1) with the domain $\Omega := (0,1)^2$ and the right hand side

$$f(x,y,t) = x(1-x)y(1-y)e^t + 2[y(1-y) + x(1-x)]e^t,$$

which has been derived from the prescribed exact solution

$$u(x,y,t) = x(1-x)y(1-y)e^t.$$

The initial data is $u_0(x,y) = u(x,y,0)$.

This and all other examples have the property that there is no spatial error which can be seen as follows. For each fixed time t, the exact solution $u(t)$ is an element of the finite element space V_h. Therefore, the standard semi-discrete solution with respect to space $u_h(t) \in V_h$ is equal to $u(t)$. If we now apply the time discretization to $u_h(t) = u(t)$ we get $u_{h,\tau}(t) = u_\tau(t)$. Since space and time discretizations are of Galerkin type we obtain $u_{\tau,h}(t) = u_{h,\tau}(t) = u_\tau(t)$. Therefore, in all of our examples, the full discretization error $u(t) - u_{\tau,h}(t)$ is equal to the time discretization error $u(t) - u_\tau(t)$, i.e. we will concentrate only on the time discretization error and exclude interactions with the spatial error.

We apply the time discretization schemes cGP(1), cGP(2) and dG(1) with an equidistant time step size $\tau = T/N$. To measure the error, the following discrete time L^∞-norm of a function $v : I \to L^2(\Omega)$ is used

$$\|v\|_\infty := \max_{1 \le n \le N} \|v^-(t_n)\|_{L^2(\Omega)}, \qquad v^-(t_n) := \lim_{t \to t_n - 0} v(t), \qquad t_n := n\tau.$$

The behavior of the standard L^2-norm $\|\cdot\|_2 := \|\cdot\|_{L^2(I,L^2(\Omega))}$ and the discrete L^∞-norm of the time discretization error $u(t) - u_\tau(t)$ over the time interval $I = [0, 1]$ can be seen in Table 1 and 2, respectively. The estimated value of the experimental order of convergence (EOC) is also calculated and compared with the theoretical order of convergence.

	cGP(1)		cGP(2)		dG(1)	
$1/\tau$	$\|u - u_\tau\|_2$	EOC	$\|u - u_\tau\|_2$	EOC	$\|u - u_\tau\|_2$	EOC
10	5.65E-05		3.04E-07		3.08E-05	
20	1.41E-05	2.00	3.64E-08	3.06	8.28E-06	1.90
40	3.53E-06	2.00	4.50E-09	3.02	2.16E-06	1.94
80	8.83E-07	2.00	5.60E-10	3.00	5.53E-07	1.97
160	2.21E-07	2.00	7.00E-11	3.00	1.40E-07	1.98
320	5.52E-08	2.00	8.75E-12	3.00	3.52E-08	1.99
640	1.38E-08	2.00	1.09E-12	3.00	8.82E-09	2.00
1280	3.45E-09	2.00			2.21E-09	2.00
2560	8.63E-10	2.00			5.53E-10	2.00
5120	2.16E-10	2.00			1.38E-10	2.00
10240	5.39E-11	2.00			3.46E-11	2.00
20480	1.35E-11	2.00			8.64E-12	2.00
40960	3.37E-12	2.00			2.16E-12	2.00
81920	8.37E-13	2.01			5.42E-13	2.00

Table 1: Error norms $\|u - u_\tau\|_2$ for the first test case.

We see that the cGP(2)-method is of order 3 in the L^2-norm and superconvergent of order 4 at the discrete points t_n, while the dG(1)-method is of order 2 in the L^2-norm and superconvergent of order 3 at the end points of the time intervals as expected from the theory. The cGP(1)-method is of order 2 everywhere which is the same behavior as that of the well-known Crank-Nicolson scheme.

Next, we perform numerical tests to analyze the corresponding behavior of the multigrid solver for the different time discretization schemes. As explained before, the solver uses the Symmetric Successive Overrelaxation (SSOR) method in the smoothing process and applies one pre- and post-smoothing step. We present the average number of multigrid and preconditioned BiCGStab iterations per time step for solving the corresponding systems in Table 3 and 4. For a better comparison, the BiCGStab solver also utilizes SSOR as a preconditioner. 'Lev' denotes the refinement level of the space mesh. From Table 3, we see that the multigrid solver requires almost the same number of iterations for the different presented time

	cGP(1)		cGP(2)		dG(1)	
$1/\tau$	$\|u-u_\tau\|_\infty$	EOC	$\|u-u_\tau\|_\infty$	EOC	$\|u-u_\tau\|_\infty$	EOC
10	3.63E-06		4.14E-07		1.80E-05	
20	9.09E-07	2.00	2.65E-08	3.97	2.59E-06	2.80
40	2.27E-07	2.00	1.67E-09	3.99	3.51E-07	2.88
80	5.68E-08	2.00	1.05E-10	3.99	4.59E-08	2.93
160	1.42E-08	2.00	6.57E-12	4.00	5.90E-09	2.96
320	3.55E-09	2.00	4.01E-13	4.03	7.49E-10	2.98
640	8.88E-10	2.00			9.45E-11	2.99
1280	2.22E-10	2.00			1.19E-11	2.99
2560	5.55E-11	2.00			1.33E-12	3.16
5120	1.39E-11	2.00				
10240	3.46E-12	2.00				
20480	8.62E-13	2.00				

Table 2: Error norms $\|u-u_\tau\|_\infty$ for the first test case with known analytical solution.

Lev	$\tau = 1/20$	$\tau = 1/80$	$\tau = 1/320$	$\tau = 1/1280$
6	13-13-13	13-13-13	12-12-12	10-10-10
7	13-13-13	13-13-13	13-13-13	12-12-12
8	14-13-14	13-13-13	13-13-13	13-13-13

Table 3: Averaged multigrid iterations per time step for cGP(1) - cGP(2) - dG(1).

Lev	$\tau = 1/20$	$\tau = 1/80$	$\tau = 1/320$	$\tau = 1/1280$
3	7-10-10	5-13-12	4-17-14	5-15-16
4	14-18-19	10-14-16	5-15-15	4-15-16
5	29-35-36	22-25-29	12-16-19	5-15-16
6	61-70-76	43-48-53	23-24-32	12-16-19
7	108-138-158	81-89-101	46-50-60	26-27-33
8	214-284-327	160-168-201	88-100-119	52-50-63

Table 4: Averaged BiCGStab iterations per time step for cGP(1) - cGP(2) - dG(1).

discretization schemes. Moreover, the number of multigrid iterations remains fairly constant if we increase the refinement level of the space mesh. There is also no noticeable increase in the number of iterations if we decrease the time step by a factor of 2. This means that the behavior of the multigrid solver is almost independent of the space mesh size and the time step. On the other hand, Table 4 shows that the averaged number of iterations per time step increases almost by a factor of 2 if we increase the space mesh level in the BiCGStab solver.

Next, in order to measure and compare the efficiency of the two iterative solvers for the different time schemes, we present in Tables 5-7 the averaged CPU-time required for one solver iteration on a given space mesh level. Both solvers, the multigrid and BiCGStab method,

$1/\tau$	Multigrid Solver			BiCGStab Solver		
	cGP(1)	**cGP(2)**	**dG(1)**	**cGP(1)**	**cGP(2)**	**dG(1)**
	Sec	Sec	Sec	Sec	Sec	Sec
20	0.014	0.024	0.030	0.003	0.008	0.007
80	0.014	0.032	0.029	0.003	0.008	0.012
320	0.012	0.032	0.030	0.005	0.011	0.013
1280	0.014	0.026	0.036	0.007	0.012	0.018

Table 5: CPU-time per solver iteration for space mesh level=6.

$1/\tau$	Multigrid Solver			BiCGStab Solver		
	cGP(1)	**cGP(2)**	**dG(1)**	**cGP(1)**	**cGP(2)**	**dG(1)**
	Sec	Sec	Sec	Sec	Sec	Sec
20	0.046	0.101	0.100	0.013	0.039	0.033
80	0.048	0.105	0.105	0.013	0.034	0.038
320	0.044	0.132	0.133	0.018	0.044	0.038
1280	0.048	0.145	0.105	0.022	0.053	0.048

Table 6: CPU-time per solver iteration for space mesh level=7.

respectively, have been implemented on our solver package FEAT2 (www.featflow.de). The CPU-times have been measured on an AMD Opteron 250 double CPU at 2.4GHz. In Table 5, the CPU-times in seconds are related to space mesh level 6. We observe that the CPU-time in case of cGP(2) or dG(1) is almost 2-3 times the CPU-time of cGP(1) for both, the multigrid and BiCGStab solver. We also note that the CPU-time grows approximately by a factor of 4-5 if we increase the space mesh level to level 7 and 8 in Table 6 and 7. These factors are nearly optimal since the number of space unknowns is increased by a factor of 4 if the level is increased by one.

$1/\tau$	Multigrid Solver			BiCGStab Solver		
	cGP(1)	**cGP(2)**	**dG(1)**	**cGP(1)**	**cGP(2)**	**dG(1)**
	Sec	Sec	Sec	Sec	Sec	Sec
20	0.192	0.475	0.575	0.064	0.199	0.186
80	0.191	0.568	0.487	0.065	0.177	0.189
320	0.224	0.538	0.532	0.073	0.175	0.204
1280	0.214	0.701	0.702	0.086	0.191	0.175

Table 7: CPU-time per solver iteration for space mesh level=8.

Next we compare the time discretization schemes with respect to accuracy and numerical costs. Here, the multigrid solver uses four SSOR iterations in the pre- and post-smoothing step. Table 8 shows, for different sizes of the time step τ and different time discretization

	cGP(1)		cGP(2)		dG(1)	
$1/\tau$	$\|u-u_\tau\|_2$	CPU	$\|u-u_\tau\|_2$	CPU	$\|u-u_\tau\|_2$	CPU
10	5.65E-05	2	3.04E-07	4	3.08E-05	4
20	1.41E-05	3	3.64E-08	8	8.28E-06	8
40	3.53E-06	6	4.50E-09	16	2.16E-06	17
80	8.83E-07	11	5.60E-10	28	5.53E-07	29
160	2.21E-07	23	7.00E-11	61	1.40E-07	59
320	5.52E-08	47	8.75E-12	115	3.52E-08	117
640	1.38E-08	87	1.09E-12	199	8.82E-09	203
1280	3.45E-09	171			2.21E-09	399
2560	8.63E-10	298			5.53E-10	839
5120	2.16E-10	517			1.38E-10	1483
10240	5.39E-11	825			3.46E-11	3160
20480	1.35E-11	1677			8.64E-12	6662
40960	3.37E-12	4131			2.16E-12	13519
81920	8.37E-13	10568			5.42E-13	27513

Table 8: Error norms $\|u-u_\tau\|_2$ and total CPU-times to achieve the accuracy of 10^{-12}.

schemes, the global L^2-norm error and the total CPU-time required for the computations in all time intervals. The space discretization was done on mesh level 6. One can see that, in order to achieve the accuracy of 10^{-12}, we need the very small time step $\tau = 1/81920$ for the cGP(1) and dG(1) scheme while this accuracy can be already achieved with $\tau = 1/640$ in the cGP(2) scheme. To compare the numerical costs per time step let us note that the number

of multigrid iterations to solve one linear block system is approximately the same (about 5) for the three time discretization schemes. However, the cost of one multigrid iteration in the cGP(2) or dG(1) method is about 2-3 times higher than in cGP(1). Nevertheless, for a desired accuracy of 10^{-12}, the cGP(2) scheme is about 50 times faster than cGP(1) due to the much larger time step size required for cGP(2).

In Table 9, we show the analogous comparison between the three time discretizations with respect to the numerical costs and the accuracy measured in the discrete L^∞-norm. Due to its superconvergence of order 3 in the discrete time points, the dG(1)-method is now faster than cGP(1) which is only of order 2. The most efficient scheme is again cGP(2).

	cGP(1)		cGP(2)		dG(1)	
$1/\tau$	$\|u-u_\tau\|_2$	CPU	$\|u-u_\tau\|_2$	CPU	$\|u-u_\tau\|_2$	CPU
10	3.63E-06	2	4.14E-07	4	1.80E-05	4
20	9.09E-07	3	2.65E-08	8	2.59E-06	8
40	2.27E-07	6	1.67E-09	16	3.51E-07	17
80	5.68E-08	11	1.05E-10	28	4.59E-08	29
160	1.42E-08	23	6.57E-12	61	5.90E-09	59
320	3.55E-09	47	4.01E-13	115	7.49E-10	117
640	8.88E-10	87			9.45E-11	203
1280	2.22E-10	171			1.19E-11	399
2560	5.55E-11	298			1.33E-12	839
5120	1.39E-11	517				
10240	3.46E-12	825				
20480	8.62E-13	1677				

Table 9: Error norms $\|u-u_\tau\|_\infty$ and total CPU-time to achieve the accuracy of 10^{-12}.

To show that the proposed time discretization schemes can also efficiently handle the case when the solution approaches a steady state, we provide numerical tests with very large time steps. We consider problem (1) with $\Omega = (0,1)^2$ and the prescribed (time-independent, steady state) solution

$$u(x,y,t) := x(1-x)y(1-y).$$

For this function u, we compute the corresponding right hand side f. As initial data we take $u_0 = 0$. Table 10 and 11 indicate for the multigrid and the BiCGStab method, respectively, the number of solver iterations required for one time step. From Table 10, one can see that there is no big difference in the number of solver iterations for time step size $\tau = 10^{-6}$ up to $\tau = 10^6$. This means that the behavior of the multigrid convergence is pretty robust with

Lev	$\tau = 10^{-6}$	$\tau = 10^{-3}$	$\tau = 1$	$\tau = 10^{3}$	$\tau = 10^{6}$
6	3-5-6	4-4-5	5-6-6	5-6-6	5-6-6
7	3-5-6	5-5-6	6-6-6	5-6-6	5-6-6
8	2-4-5	5-6-6	5-6-6	6-6-6	6-6-6

Table 10: Averaged multigrid iterations per time step for cGP(1) - cGP(2) - dG(1).

Lev	$\tau = 10^{-6}$	$\tau = 10^{-3}$	$\tau = 1$	$\tau = 10^{3}$	$\tau = 10^{6}$
6	8-17-19	14-16-18	78-78-82	69-83-82	72-86-86
7	7-17-17	31-26-35	145-173-172	131-147-164	149-170-157
8	6-15-16	60-51-67	280-362-297	280-290-290	288-322-293

Table 11: Averaged BiCGStab iterations per time step for cGP(1) - cGP(2) - dG(1).

respect to very small and very large time steps. However, from Table 11, we observe that, in the case of time step sizes $\tau \geq 10^{-3}$, the BiCGStab solver is suitable only for lower space mesh levels since the number of iterations nearly doubles if we increase the space mesh level by one. Furthermore, the number of BiCGStab iterations grows on each space mesh level if we increase the size of the time step in the range between $\tau = 10^{-6}$ and $\tau = 1$.

At the end, we want to show the behavior of the presented time discretizations for another prototypical problem which is more oscillating in time. For this, in problem (1) with $\Omega = (0,1)^2$, we prescribe the exact solution

$$u(x,y,t) := x(1-x)y(1-y)\sin(10\pi t)$$

and compute the corresponding right hand side f and the initial data $u_0(x,y) = u(x,y,0)$. Table 12 shows the behavior of the discrete L^∞-norm of the error and the EOC over the time interval $[0,1]$. One can see that the presented time discretizations confirm their theoretical order of convergence also for such an oscillatory solution. In order to compare the three discretization schemes, one can state that the cGP(2) gains the same accuracy at an eight times larger time step size than dG(1) while for cGP(1), we actually need to bisect the time step size ten times more. Consequently, the cGP(2)-method is almost thousand times and dG(1) almost hundred times faster than cGP(1) for such an oscillatory exact solution. This effect has been much more visible when we considered the more complex solution with a higher number of oscillations in the sine function, i.e., if we have prescribed the solution

$$u(x,y,t) := x(1-x)y(1-y)\sin(100\pi t)$$

in our model problem. Here, the discrepancy has been even more obvious between the high order schemes dG(1) and cGP(2) on the one side and the second order method cGP(1), which

	cGP(1)		cGP(2)		dG(1)	
$1/\tau$	$\|u-u_\tau\|_\infty$	EOC	$\|u-u_\tau\|_\infty$	EOC	$\|u-u_\tau\|_\infty$	EOC
20	5.85E-03		2.03E-04		4.19E-04	
40	1.50E-03	1.96	1.31E-05	3.95	7.75E-05	2.44
80	3.72E-04	2.01	8.34E-07	3.97	1.06E-05	2.87
160	9.43E-05	1.98	5.29E-08	3.98	1.40E-06	2.92
320	2.35E-05	2.00	3.32E-09	3.99	1.80E-07	2.96
640	5.88E-06	2.00	2.08E-10	4.00	2.29E-08	2.97
1280	1.47E-06	2.00	1.30E-11	4.00	2.90E-09	2.98
2560	3.68E-07	2.00	8.13E-13	4.00	3.65E-10	2.99
5120	9.19E-08	2.00			4.57E-11	2.99
10240	2.30E-08	2.00			5.73E-12	3.00
20480	5.75E-09	2.00			7.14E-13	3.00
40960	1.44E-09	2.00				
81920	3.59E-10	2.00				
163840	8.97E-11	2.00				
327680	2.24E-11	2.00				
655360	5.61E-12	2.00				
1310720	1.40E-12	2.00				
2621440	3.50E-13	2.00				

Table 12: Error norms $\|u-u_\tau\|_\infty$ for the "sin-test" case.

is close to the standard Crank-Nicolson scheme and representative for many other 2nd order schemes, on the other side. These tests demonstrate the superior quality of the high order approaches, together with corresponding fast solvers, for complex dynamical problems.

7 Conclusion

We have presented continuous Galerkin-Petrov and discontinuous Galerkin time discretization schemes for the two dimensional heat equation where the spatial discretization has been performed by means of biquadratic finite elements. The associated linear systems have been solved using a preconditioned Krylov-space and a geometrical multigrid method. Firstly, from the numerical studies, we observe that the estimated experimental orders of convergence confirm the theoretical orders. Furthermore, the tests show that the cGP(2)-scheme provides significantly more accurate numerical solutions than the other presented schemes cGP(1) and dG(1) which means that quite large time step sizes are allowed to gain highly accurate

results. Secondly, we can see that the used multigrid solver is much more efficient than the preconditioned BiCGStab solver since it shows a robust convergence behavior which is nearly independent of the space mesh size and the time step such that large time steps get feasible also with respect to efficient solution methods. In our future work, we plan to extend these time discretization schemes to the Navier-Stokes equations to simulate complex time dependent flow problems in a very efficient way together with highly sophisticated multigrid-FEM techniques for incompressible saddle point problems

References

[1] A. K. Aziz and Peter Monk. Continuous finite elements in space and time for the heat equation. *Math. Comp.*, 52(186):255–274, 1989.

[2] W. Hackbusch. *Multi-Grid Methods and Applications.* Springer-Verlag, Berlin, 1985.

[3] M. Köster and S. Turek. The influence of higher order FEM discretisations on multigrid convergence. *Computational Methods in Applied Mathematics*, 6(2):221–232, 2006.

[4] F. Schieweck. A-stable discontinuous Galerkin-Petrov time discretization of higher order. *J. Numer. Math.*, 18(1):25 – 57, 2010.

[5] Vidar Thomée. *Galerkin finite element methods for parabolic problems*, volume 25 of *Springer Series in Computational Mathematics*. Springer-Verlag, Berlin, second edition, 2006.

[6] S. Turek. *Efficient solvers for incompressible flow problems. An algorithmic and computational approach*, volume 6 of *Lecture Notes in Computational Science and Engineering*. Springer, 1999.