Available online at www.sciencedirect.com
ScienceDirect
Journal of Computational Physics 218 (2006) 295–323
JOURNAL OF COMPUTATIONAL PHYSICS
www.elsevier.com/locate/jcp

Discontinuous Galerkin method based on non-polynomial approximation spaces ☆

Ling Yuan, Chi-Wang Shu *

Division of Applied Mathematics, Brown University, Box F, Providence, RI 02912, United States

Received 15 November 2005; received in revised form 10 February 2006; accepted 13 February 2006
Available online 19 April 2006

Abstract

In this paper, we develop discontinuous Galerkin (DG) methods based on non-polynomial approximation spaces for numerically solving time dependent hyperbolic and parabolic and steady state hyperbolic and elliptic partial differential equations (PDEs). The algorithm is based on approximation spaces consisting of non-polynomial elementary functions such as exponential functions, trigonometric functions, etc., with the objective of obtaining better approximations for specific types of PDEs and initial and boundary conditions. It is shown that L^2 stability and error estimates can be obtained when the approximation space is suitably selected. It is also shown with numerical examples that a careful selection of the approximation space to fit individual PDE and initial and boundary conditions often provides more accurate results than the DG methods based on the polynomial approximation spaces of the same order of accuracy.

Keywords: Discontinuous Galerkin method; Approximation space; Exponential functions; Trigonometric functions; Error estimate

1. Introduction

When the discontinuous Galerkin (DG) method [11] is used to solve partial differential equations (PDEs), the piecewise polynomial space is the commonly chosen finite element approximation space. However, for some PDEs and initial/boundary conditions, piecewise polynomials may not provide the best approximation to the solution. A major advantage of the DG method is its flexibility with the finite element approximation space. Essentially any linear space can be used as the local approximation space, and the approximation space can vary from element to element and also for different time t. This flexibility comes from the fact that we do not need to enforce any continuity at the element interfaces. Traditional continuous finite element methods certainly do not have this flexibility. In this paper, we explore this flexibility and propose the use of spaces based on non-polynomial elementary functions such as exponential functions, trigonometric functions etc., with the objective of obtaining better approximations to specific types of PDEs and initial/boundary

☆ Research supported by ARO grant W911NF-04-1-0291, NSF grant DMS-0510345 and AFOSR grant FA9550-05-1-0123.
* Corresponding author. Tel.: +1 401 863 2549; fax: +1 401 863 1355.
E-mail addresses: lyuan@dam.brown.edu (L. Yuan), shu@dam.brown.edu (C.-W. Shu).

0021-9991/$ - see front matter © 2006 Elsevier Inc. All rights reserved.
doi:10.1016/j.jcp.2006.02.013

conditions. It is shown that as long as the approximation space is suitably selected, we can obtain similar L^2 stability and error estimates as for the piecewise polynomial space.

We now mention some related work in the literature. In [5], Cockburn et al. propose the use of the locally divergence-free polynomial space in the DG method to solve the Maxwell equations and they achieve better results compared to the DG method based on the classical piecewise polynomial space P^k, see also related work in [16–18]. However, the locally divergence-free polynomial space is still based on polynomials. In this paper we intend to focus on non-polynomial spaces in the DG method. Another line of related work is the solution of singular perturbation problems by using exponentially fitted schemes, see for example the work of Kadalbajoo and Patidar [15] and of Reddy and Chakravarthy [19]. We also refer to [2] in which non-polynomial spaces are used in local essentially non-oscillatory (ENO) reconstructions for solving hyperbolic conservation laws.

The boundary layer and highly oscillatory problems are examples that the polynomial space does not make a good approximation if the mesh is coarse. For the boundary layer problems, the slope of the solution near the boundary is very large (Fig. 1.1, left), which is better approximated by exponential functions rather than by polynomials. For the highly oscillatory problems (Fig. 1.1, right), the solution is better approximated by trigonometric functions. It is intuitive that an exponential-function space should be used for good approximation to solutions with boundary layers, and a trigonometric-function space should be used to the highly oscillatory problems. Based on this intuition, we identified suitable approximation spaces, and studied L^2 stability for the DG method based on these spaces. It can also be proven that, if the exact solution is smooth enough, the rate of convergence is the same as the dimension of the local approximation space. Similar results can be obtained when the DG method based on these two spaces is applied to other problems.

In the proposed *modified DG method*, rather than keeping the approximation spaces fixed, we allow time-varying approximation spaces. These spaces are indexed by parameters and the parameters are adjusted at each time-step according to the numerical solution of the last time-step. We propose methods to automatically adjust the parameters so that the approximation spaces can better fit the underlying solution. Numerical tests indicate that these methods do find the best-fitting parameters and can yield better results compared to the polynomial approximation space.

This paper is organized as follows. In Section 2, we give a brief review about the DG method. In Section 3, approximation spaces different from the polynomial space are introduced. Criteria for selecting suitable approximation spaces are given, and approximation results are presented for spaces satisfying these criteria. In Section 4, the method of using time-varying approximation spaces in the DG method, which is called the *modified DG method*, is proposed, and two methods for adjusting the parameters in the basis functions of the approximation spaces are presented. In the same section, theoretical results of L^2-stability as well as error estimates of the proposed DG method based on non-polynomial approximation spaces are presented. Section 5

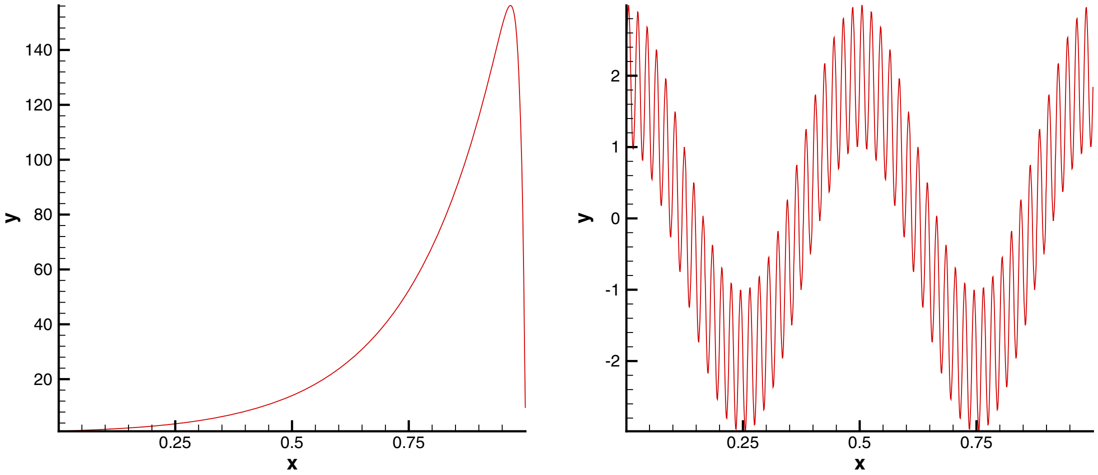

Fig. 1.1. Examples of solutions to the boundary layer problem (left) and the highly oscillatory problem (right).

consists of numerical results showing the accuracy of the *modified DG method*. Concluding remarks are given in Section 6. In Appendix A, we give the verification that some of the approximation spaces used in this paper do satisfy the criteria required in the proof of the approximation results in Section 3.

2. The DG method

We will only briefly review the DG method here for solving hyperbolic problems. First, consider the one-dimensional conservation law

$$u_t + f(u)_x = 0. \tag{2.1}$$

We denote the mesh by $I_j = [x_{j-\frac{1}{2}}, x_{j+\frac{1}{2}}]$ for $j = 1, \ldots, N$, with the center of the cell denoted by $x_j = \frac{1}{2}(x_{j-\frac{1}{2}} + x_{j+\frac{1}{2}})$ and the size of each cell by $\Delta x_j = x_{j+\frac{1}{2}} - x_{j-\frac{1}{2}}$. We will denote $\Delta x = \max_j \Delta x_j$ and will assume a regular mesh which has a bounded ratio between the largest and the smallest Δx_j. If we multiply (2.1) by an arbitrary test function $v(x)$, integrate over the interval I_j, and integrate by parts, we obtain the weak formulation

$$\int_{I_j} u_t v \, dx - \int_{I_j} f(u) v_x \, dx + f(u_{j+\frac{1}{2}}) v_{j+\frac{1}{2}} - f(u_{j-\frac{1}{2}}) v_{j-\frac{1}{2}} = 0. \tag{2.2}$$

Here $u_{j+\frac{1}{2}} = u(x_{j+\frac{1}{2}})$. We replace both the solution u and the test function v by U and V, which are in the approximation space V_h. For standard DG method, V_h is taken as the space of piecewise polynomials

$$V_h = \{v : v|_{I_j} \in P^k(I_j), j = 1, \ldots, N\}, \tag{2.3}$$

where $P^k(I)$ denotes the space of polynomials in an interval I of degree at most k.

The DG method is then given by the following: Find $U(\cdot, t) \in V_h$ such that

$$\int_{I_j} U_t V \, dx - \int_{I_j} f(U) V_x \, dx + \widehat{f(U)}_{j+\frac{1}{2}} V^-_{j+\frac{1}{2}} - \widehat{f(U)}_{j-\frac{1}{2}} V^+_{j-\frac{1}{2}} = 0 \tag{2.4}$$

for all test functions $V \in V_h$. The "numerical flux" $\widehat{f(U)}_{j+\frac{1}{2}} = \hat{f}(U^-_{j+\frac{1}{2}}, U^+_{j+\frac{1}{2}})$ is chosen to be a monotone flux for the scalar case, and to be a numerical flux based on exact or approximate Riemann solvers for the system case. Notice that we use V^- and V^+ to denote the left and right limits of V, respectively, at the interface where V is discontinuous. The resulting method of lines ODE is then discretized by the nonlinearly stable high order TVD Runge–Kutta methods in [20,12].

For more details of the DG method, we refer the reader to the series of papers of Cockburn and Shu [7–9], Cockburn et al. [6,4], the lecture notes [3], and the review paper [11].

For the convection–diffusion problems

$$u_t + f(u)_x = (a(u, x) u_x)_x, \tag{2.5}$$

we use the local discontinuous Galerkin (LDG) method [10]. In the LDG method, local auxiliary variables are introduced and second order derivatives are converted to first order derivatives using those auxiliary variables which represent first order derivatives of the solution. The usual DG procedure is then applied and suitable numerical fluxes are chosen for stability and convergence of the scheme. The auxiliary variables can be locally eliminated thus they do not increase the size of the numerical system or pose global storage problem. We will not review the details of the LDG method here to save space and refer the reader to the papers of Cockburn and Shu [10,11].

3. Non-polynomial approximation spaces

In the standard DG method, the piecewise polynomial space (2.3) is used as the finite element space (both the trial space and the test space). However, the DG method does provide the flexibility of using other non-polynomial finite element spaces. The motivation to use non-polynomial finite element spaces is to obtain better approximations for specific solutions of PDEs, such as the boundary layer solutions and oscillatory solutions in Fig. 1.1, for which exponential/trigonometric functions instead of polynomials as basis functions for the new approximation spaces are expected to yield better numerical results because these functions themselves can approximate the exact solutions better than polynomials. However, some criteria are needed for the choice of basis functions of the non-polynomial finite element spaces, in order for those spaces to have the same approximation rates as those of polynomial finite element spaces of the same dimension.

3.1. One-dimensional case

We first consider the one-dimensional case. The following are examples of possible non-polynomial finite element spaces in one dimension.

- The exponential space I:

$$E^k(\alpha)=\left\{v: v|_{I_j}\in \operatorname{span}\{e^{\alpha_j(x-x_j)}, e^{\alpha_j(x-x_j)}(x-x_j),\ldots,e^{\alpha_j(x-x_j)}(x-x_j)^k\}, x\in I_j\right\}. \tag{3.1}$$

- The exponential space II:

$$\bar{E}^k(\alpha)=\left\{v: v|_{I_j}\in \operatorname{span}\{e^{\alpha_j(x-x_j)}, (x-x_j),\ldots,(x-x_j)^k\}, x\in I_j\right\}. \tag{3.2}$$

- The trigonometric polynomial space:

$$\begin{aligned}T^1(\alpha)&=\left\{v: v|_{I_j}\in \operatorname{span}\{1,\sin\alpha_j(x-x_j)\}, x\in I_j\right\},\\ T^2(\alpha)&=\left\{v: v|_{I_j}\in \operatorname{span}\{1,\sin\alpha_j(x-x_j),\cos\alpha_j(x-x_j)\}, x\in I_j\right\}\end{aligned} \tag{3.3}$$

with apparent definition for the general $T^k(\alpha)$ space.

We would like to choose non-polynomial finite element spaces which have the same approximation rates as those of polynomial finite element spaces of the same dimension. The criteria to achieve this purpose are summarized in the following proposition.

Proposition 3.1. *Assume $\{v_0, v_1, \ldots, v_k\}$ is a local basis of the space V_h in cell I_j. If there are constants a_{il} and b_i independent of Δx_j and satisfying $a_{ii}\neq 0$ such that*

$$\left|v_i(x)-\sum_{l=i}^{k}a_{il}(x-x_j)^l\right|\leqslant b_i(\Delta x_j)^{k+1}\quad \forall x\in I_j, i=0,\ldots,k, \tag{3.4}$$

then for any function $u(x)\in H^{k+1}(I_j)$, there exist $v_h\in V_h$ and a constant C independent of Δx_j such that

$$|v_h(x)-u(x)|\leqslant C\|u\|_{H^{k+1}(I_j)}(\Delta x_j)^{k+1/2}\quad \forall x\in I_j. \tag{3.5}$$

Proof. Let T be the Taylor expansion operator at the point x_j into the standard piecewise polynomial space P^k. We have

$$Tu(x)=\sum_{l=0}^{k}c_l(x-x_j)^l,$$

where $c_l=\frac{1}{l!}u^{(l)}(x_j)$. Let

$$A=\begin{pmatrix}a_{00} & a_{01} & \cdots & a_{0k}\\ 0 & a_{11} & \cdots & a_{1k}\\ \cdots & \cdots & \cdots & \cdots\\ 0 & 0 & \cdots & a_{kk}\end{pmatrix},\quad b=\begin{pmatrix}b_0\\ b_1\\ \cdots\\ b_k\end{pmatrix},\quad c=\begin{pmatrix}c_0\\ c_1\\ \cdots\\ c_k\end{pmatrix},\quad p=\begin{pmatrix}1\\ x-x_j\\ \cdots\\ (x-x_j)^k\end{pmatrix}.$$

Let $f(x)=\sum_{i=0}^{k}(u^{(i)}(x))^2$, we have

$$\left|\int_{I_j} f(x)\mathrm{d}x-\Delta x_j f(x_j)\right| = \left|\int_{I_j}\int_{x_j}^{x} f'(t)\mathrm{d}t\,\mathrm{d}x\right| \leqslant \Delta x_j\int_{I_j}|f'(t)|\mathrm{d}t$$

and also

$$\int_{I_j}|f'(x)|\mathrm{d}x \leqslant 2\sum_{i=0}^{k}\int_{I_j}|u^{(i)}(x)u^{(i+1)}(x)|\mathrm{d}x \leqslant \sum_{i=0}^{k}\int_{I_j}(u^{(i)}(x))^2\,\mathrm{d}x+\sum_{i=0}^{k}\int_{I_j}(u^{(i+1)})^2\,\mathrm{d}x \leqslant 2\|u\|^2_{H^{k+1}(I_j)}.$$

Therefore

$$\begin{aligned}\Delta x_j c^{\mathrm{T}}c &= \Delta x_j\sum_{i=0}^{k}\left(\frac{1}{i!}u^{(i)}(x_j)\right)^2 \leqslant \Delta x_j\sum_{i=0}^{k}(u^{(i)}(x_j))^2 = \Delta x_j f(x_j) \leqslant \int_{I_j} f(x)\mathrm{d}x + \left|\int_{I_j} f(x)\mathrm{d}x-\Delta x_j f(x_j)\right|\\ &\leqslant (1+2\Delta x_j)\|u\|^2_{H^{k+1}(I_j)}.\end{aligned}$$

Now let $d=(A^{\mathrm{T}})^{-1}c$ and $v_h=d^{\mathrm{T}}v$. We have

$$\begin{aligned}|v_h-Tu| &= |d^{\mathrm{T}}v-c^{\mathrm{T}}p| = |c^{\mathrm{T}}A^{-1}(v-Ap)| \leqslant \sqrt{c^{\mathrm{T}}c}\|A^{-1}\|\|v-Ap\|\\ &\leqslant \sqrt{\frac{1}{\Delta x_j}(1+2\Delta x_j)\|u\|^2_{H^{k+1}(I_j)}}\|A^{-1}\|\|b\|(\Delta x_j)^{k+1} \leqslant C\|u\|_{H^{k+1}(I_j)}(\Delta x_j)^{k+1/2},\end{aligned}$$

where in the second last inequality above we have used the assumption (3.4). The unidentified norms are all L^2 norms. C here and below is a generic constant independent of u and Δx, which may not have the same value at different locations.

We also notice that

$$\begin{aligned}\|u-Tu\|_{L^\infty(I_j)} &= \sup_{x\in I_j}\left|\int_{x_j}^{x}u^{(k+1)}(t)\frac{(x-t)^k}{k!}\mathrm{d}t\right| \leqslant \sup_{x\in I_j}\left(\int_{x_j}^{x}|u^{(k+1)}(t)|^2\mathrm{d}t\right)^{1/2}\left(\int_{x_j}^{x}\left|\frac{(x-t)^k}{k!}\right|^2\mathrm{d}t\right)^{1/2}\\ &\leqslant C|u|_{H^{k+1}(I_j)}(\Delta x_j)^{k+1/2}.\end{aligned}$$

Finally we have, for all $x\in I_j$,

$$\begin{aligned}|u(x)-v_h(x)| &\leqslant |u(x)-Tu(x)|+|v_h(x)-Tu(x)| \leqslant C|u|_{H^{k+1}(I_j)}(\Delta x_j)^{k+1/2}+C\|u\|_{H^{k+1}(I_j)}(\Delta x_j)^{k+1/2}\\ &\leqslant C\|u\|_{H^{k+1}(I_j)}(\Delta x_j)^{k+1/2}\end{aligned}$$

and the proof is now complete. □

Next, we estimate the approximation rate in the L^2 norm.

Proposition 3.2. *Assume V_h is a space satisfying the condition (3.4) in Proposition 3.1 in each cell $I_j\in\Omega$. Let P_h be the L^2 projection operator into the space V_h. For any function $u(x)\in H^{k+1}(\Omega)$, there exists a constant C such that*:

$$\|P_h u-u\|_{L^2(\Omega)} \leqslant C\|u\|_{H^{k+1}(\Omega)}(\Delta x)^{k+1}. \tag{3.6}$$

Proof. We choose the same v_h as that in Proposition 3.1. Squaring both sides of (3.5) and then integrating in the cell I_j, we obtain

$$\|u-v_h\|^2_{L^2(I_j)} \leqslant C\|u\|^2_{H^{k+1}(I_j)}(\Delta x_j)^{2k+2}\quad \forall j.$$

Therefore,

$$\|P_h u - u\|^2_{L^2(\Omega)} \leqslant \|u - v_h\|^2_{L^2(\Omega)} = \sum_j \|u - v_h\|^2_{L^2(I_j)} \leqslant C \sum_j \|u\|^2_{H^{k+1}(I_j)} (\Delta x_j)^{2k+2} \leqslant C\|u\|^2_{H^{k+1}(\Omega)} (\Delta x)^{2k+2}.$$

Taking square roots on both sides finishes the proof. □

We verify in Appendix A that the three spaces (3.1)–(3.3) satisfy the condition (3.4) in Proposition 3.1.

We now show numerical results in Tables 3.1 and 3.2 for the approximations to $u(x) = \mathrm{e}^x$ and to $u(x) = \sin(x)$ respectively. From the results in these two tables, we can see that we obtain the optimal order of the approximation rate (equal to the dimension of the local approximation space) when using the approximation spaces $E^k(1)$ and $T^k(1)$ $(k = 1, 2)$, where, e.g., $E^k(1)$ refers to $\alpha_j = 1$ for all j, to approximate general functions. It can be seen that the approximation results are comparable to those obtained by the usual piecewise polynomial approximation for general functions. Not surprisingly, we obtain the exact solutions modulo round-off errors when approximating specific functions tailored to the specific approximation spaces.

3.2. Multi-dimensional case

In this subsection we generalize the approximation spaces to multi-dimensions. We concentrate our attention on the two-dimensional case. The following are examples of possible non-polynomial finite element spaces in two dimension.

- The exponential space:

$$\begin{aligned} E^k(\alpha,\ \beta) = \Big\{ v : v|_K \in \operatorname{span}\Big\{ &\mathrm{e}^{\alpha_K(x-x_K)+\beta_K(y-y_K)}, \mathrm{e}^{\alpha_K(x-x_K)+\beta_K(y-y_K)}(x-x_K), \mathrm{e}^{\alpha_K(x-x_K)+\beta_K(y-y_K)}(y-y_K), \\ &\mathrm{e}^{\alpha_K(x-x_K)+\beta_K(y-y_K)}(x-x_K)^2, \mathrm{e}^{\alpha_K(x-x_K)+\beta_K(y-y_K)}(x-x_K)(y-y_K), \mathrm{e}^{\alpha_K(x-x_K)+\beta_K(y-y_K)}(y-y_K)^2, \ldots, \\ &\mathrm{e}^{\alpha_K(x-x_K)+\beta_K(y-y_K)}(x-x_K)^k, \ldots, \mathrm{e}^{\alpha_K(x-x_K)+\beta_K(y-y_K)}(y-y_K)^k \Big\}, (x,y) \in K \Big\}. \end{aligned} \tag{3.7}$$

Table 3.1
L^2- and L^∞-errors of approximation to e^x $(0 \leqslant x \leqslant \pi)$. Uniform mesh with N cells

N	Exponential space $E^1(1)$				Trigonometric space $T^1(1)$			
	L^2-error		L^∞-error		L^2-error	Order	L^∞-error	Order
10	1.42E−14		2.25E−14		5.98E−02		1.38E−01	
20	9.26E−14		2.15E−13		1.50E−02	2.00	3.63E−02	1.93
40	7.64E−14		1.85E−13		3.76E−03	2.00	9.31E−03	1.96
80	6.69E−13		1.53E−12		9.39E−04	2.00	2.36E−03	1.98
160	8.37E−13		1.93E−12		2.35E−04	2.00	5.94E−04	1.99
	Exponential space $E^2(1)$				Trigonometric space $T^2(1)$			
	L^2-error		L^∞-error		L^2-error	Order	L^∞-error	Order
10	8.06E−14		1.76E−13		3.18E−03		6.35E−03	
20	5.90E−13		1.56E−12		3.96E−04	3.01	8.62E−04	2.88
40	1.10E−11		2.70E−11		4.99E−05	2.99	1.12E−04	2.94
80	1.39E−10		3.48E−10		6.24E−06	3.00	1.44E−05	2.96
160	8.78E−11		2.23E−10		7.80E−07	3.00	1.81E−06	2.99
	Polynomial space P^1				Polynomial space P^2			
	L^2-error	Order	L^∞-error	Order	L^2-error	Order	L^∞-error	Order
10	5.98E−02		1.34E−01		1.59E−03		3.26E−03	
20	1.50E−02	2.00	3.58E−02	1.90	1.99E−04	3.00	4.38E−04	2.90
40	3.76E−03	2.00	9.26E−03	1.95	2.49E−05	3.00	5.67E−05	2.95
80	9.39E−04	2.00	2.35E−03	1.98	3.12E−06	3.00	7.21E−06	2.98
160	2.35E−04	2.00	5.93E−04	1.99	3.90E−07	3.00	9.09E−07	2.99

Table 3.2
L^2- and L^∞-errors of approximation to $\sin x$ $(0 \leqslant x \leqslant \pi)$. Uniform mesh with N cells

N	Exponential space $E^1(1)$				Trigonometric space $T^1(1)$			
	L^2-error	Order	L^∞-error	Order	L^2-error	Order	L^∞-error	Order
10	9.19E−03		1.34E−02		4.60E−03		6.52E−03	
20	2.30E−03	2.00	3.33E−03	2.01	1.15E−03	2.00	1.65E−03	1.98
40	5.76E−04	2.00	8.31E−04	2.00	2.88E−04	2.00	4.13E−04	2.00
80	1.44E−04	2.00	2.07E−04	2.01	7.20E−05	2.00	1.03E−04	2.00
160	3.60E−05	2.00	5.18E−05	2.00	1.80E−05	2.00	2.58E−05	2.00
320	9.00E−06	2.00	1.30E−05	1.99	4.50E−06	2.00	6.46E−06	2.00
	Exponential space $E^2(1)$				Trigonometric space $T^2(1)$			
	L^2-error	Order	L^∞-error	Order	L^2-error		L^∞-error	
10	3.45E−04		4.76E−04		2.37E−14		1.65E−14	
20	4.32E−05	3.00	5.84E−05	3.03	3.36E−15		1.67E−15	
40	5.41E−06	3.00	7.24E−06	3.02	3.00E−13		2.21E−13	
80	6.76E−07	3.00	9.01E−07	3.01	1.52E−12		2.17E−12	
160	8.45E−08	3.00	1.12E−07	3.01	1.51E−13		1.11E−13	
320	1.07E−08	2.98	1.68E−08	2.74	2.64E−11		3.79E−11	
	Polynomial space P^1				Polynomial space P^2			
	L^2-error	Order	L^∞-error	Order	L^2-error	Order	L^∞-error	Order
10	4.60E−03		6.54E−03		1.22E−04		1.60E−04	
20	1.15E−03	2.00	1.65E−03	1.99	1.53E−05	3.00	2.02E−05	2.99
40	2.88E−04	2.00	4.13E−04	2.00	1.91E−06	3.00	2.53E−06	3.00
80	7.20E−05	2.00	1.03E−04	2.00	2.39E−07	3.00	3.17E−07	3.00
160	1.80E−05	2.00	2.58E−05	2.00	2.99E−08	3.00	3.96E−08	3.00
320	4.50E−06	2.00	6.46E−06	2.00	3.74E−09	3.00	4.95E−09	3.00

- The trigonometric polynomial space:

$$\begin{aligned}
T^1(\alpha,\ \beta) &= \{v : v|_K \in \operatorname{span}\{1, \sin\alpha_K(x - x_K), \sin\beta_K(y - y_K)\}, (x, y) \in K\} \\
T^2(\alpha,\ \beta) &= \{v : v|_K \in \operatorname{span}\{1, \sin\alpha_K(x - x_K), \sin\beta_K(y - y_K), \cos\alpha_K(x - x_K), \\
&\qquad \sin\alpha_K(x - x_K)\sin\beta_K(y - y_K), \cos\alpha_K(y - y_K)\}, (x, y) \in K\}
\end{aligned} \tag{3.8}$$

with apparent definition for the general $T^k(\alpha,\ \beta)$ space.

Here K is a two-dimensional cell with the center at the point (x_K, y_K).

Similar propositions as those in the one-dimensional case can be obtained for multi-dimensions. We will again concentrate on the two-dimensional case.

Proposition 3.3. *Assume* $\{v_{mn},\ m \geqslant 0,\ n \geqslant 0,\ m + n \leqslant k\}$ *is a local basis of the space* V_h *in cell K. If there are constants* a_{mnpq} *and* b_{mn} *independent of* $\Delta x = \operatorname{diam}(K)$ *and satisfying* $a_{mnmn} \neq 0$ *such that, for all* $x, y \in K$, $m, n \geqslant 0,\ m + n \leqslant k$,

$$\left| v_{mn} - \sum_{p \geqslant m, q \geqslant n, p+q \leqslant k} a_{mnpq}(x - x_K)^p (y - y_K)^q \right| \leqslant b_{mn}(\Delta x)^{k+1}, \tag{3.9}$$

then for any function $u \in W^{k+1,\infty}(K)$, *there exist* $v_h \in V_h$ *and a constant C independent of* Δx *such that*:

$$|v_h(x, y) - u(x, y)| \leqslant C\|u\|_{W^{k+1,\infty}(K)}(\Delta x)^{k+1} \quad \forall (x, y) \in K. \tag{3.10}$$

Proof. Let T be the Taylor expansion operator at the point (x_K, y_K) into the standard piecewise polynomial space P^k. We have

$$Tu(x,y) = \sum_{p,q\geqslant 0,p+q\leqslant k} c_{pq}(x-x_K)^p(y-y_K)^q,$$

where $c_{pq} = \frac{1}{p!q!}\partial_x^p\partial_y^q u(x_K, y_K)$.

Let the tensors $\mathrm{A}=(a_{mnpq})_{(k+1)\times(k+1)\times(k+1)\times(k+1)}$, $b=(b_{mn})_{(k+1)\times(k+1)}$, $c=(c_{pq})_{(k+1)\times(k+1)}$, $v=(v_{mn})_{(k+1)\times(k+1)}$ and $p=(p_{mn})_{(k+1)\times(k+1)}$ with $p_{mn}=(x-x_K)^m(y-y_K)^n$ for $m,n\geqslant 0$, $m+n\leqslant k$. We define the operation ":" as a product between tensors:

$$C:D=(c_{i_1i_2\ldots i_n}):(d_{j_1j_2\ldots j_m}) = \begin{cases} \left(\sum_{j_1\ldots j_m} c_{i_1\ldots i_{n-m}j_1\ldots j_m}d_{j_1\ldots j_m}\right) & \text{if } n\geqslant m,\\ \left(\sum_{i_1\ldots i_m} c_{i_1\ldots i_n}d_{i_1\ldots i_nj_{n+1}\ldots j_m}\right) & \text{otherwise.}\end{cases}$$

Notice that this is the usual inner product if the two tensors C and D are of the same order, and is the usual matrix–vector product (or vector–matrix product) if C is a matrix and D is a vector (or if C is a vector and D is a matrix). It is easy to check, just by the definition, that this operator satisfies the following associative and distributive properties:

(1) $C:(D:E)=(C:D):E$, if the order of D equals the sum of the orders of C and E;
(2) $C:(D+E)=(C:D)+(C:E)$, if D and E are of the same order.

With this definition we obtain

$$c:c=\sum_{p,q\geqslant 0,p+q\leqslant k}\left(\frac{1}{p!q!}\partial_x^p\partial_y^q u(x_i,y_j)\right)^2 \leqslant \sum_{p,q\geqslant 0,p+q\leqslant k}(\partial_x^p\partial_y^q u(x_i,y_j))^2 \leqslant \left(\sum_{p,q\geqslant 0,p+q\leqslant k}|\partial_x^p\partial_y^q u(x_i,y_j)|\right)^2$$
$$\leqslant \|u\|^2_{W^{k+1,\infty}(K)}.$$

Now we need a matrix d such that $d:A=c$. In fact, for this specific A, there exists a unique "inverse" tensor $\bar{A}$ such that $d=c:\bar{A}$. We will show how to obtain this tensor $\bar{A}$ in Appendix A. Also let $v_h=d:v$. Now we obtain

$$|v_h-Tu| = |d:v-c:p| = |d:(v-A:p)| = |c:\bar{A}:(v-A:p)| \leqslant \sqrt{c:c}\|\bar{A}\|\|v-A:p\|$$
$$\leqslant \|u\|_{W^{k+1,\infty}(K)}\|\bar{A}\|\|b\|(\Delta x)^{k+1} \leqslant C\|u\|_{W^{k+1,\infty}(K)}(\Delta x)^{k+1},$$

where $\|v-A:p\|$ and $\|b\|$ refer to the vector L^2 norm when the matrices $v-A:p$ and b are regarded as a long vector (that is, the square of the norm is the sum of squares of all the entries of the matrix), and $\|\bar{A}\|$ is the associated operator norm.

We also notice that, by the property of the Taylor expansion,

$$\|u-Tu\|_{L^\infty(K)} \leqslant C|u|_{W^{k+1,\infty}(K)}(\Delta x)^{k+1}.$$

Therefore we have, for all $(x,y)\in K$,

$$|u(x,y)-v_h(x,y)| \leqslant |u(x,y)-Tu(x,y)|+|v_h(x,y)-Tu(x,y)| \leqslant C|u|_{W^{k+1,\infty}(K)}(\Delta x)^{k+1}+C\|u\|_{W^{k+1,\infty}(K)}(\Delta x)^{k+1}$$
$$\leqslant C\|u\|_{W^{k+1,\infty}(K)}(\Delta x)^{k+1}.$$

Now the proof is complete. □

Next, we estimate the approximation rate in the L^2 norm.

Proposition 3.4. *Assume V_h is a space satisfying the condition (3.9) in Proposition 3.3 in each cell $K \in \Omega$. Let P_h be the L^2 projection operator into the space V_h. For any function $u \in W^{k+1,\infty}(\Omega)$, there exists a constant C such that*

$$\|P_h u - u\|_{L^2(\Omega)} \leqslant C\|u\|_{W^{k+1,\infty}(\Omega)}(\Delta x)^{k+1}. \tag{3.11}$$

Proof. We choose the same v_h as that in Proposition 3.3. Squaring both sides of (3.10) and then integrating in the cell K, we obtain

$$\|u - v_h\|^2_{L^2(K)} \leqslant C\|u\|^2_{W^{k+1,\infty}(K)}(\Delta x)^{2k+4} \quad \forall K.$$

Therefore,

$$\|P_h u - u\|^2_{L^2(\Omega)} \leqslant \|u - v_h\|^2_{L^2(\Omega)} = \sum_K \|u - v_h\|^2_{L^2(K)} \leqslant C\sum_K \|u\|^2_{W^{k+1,\infty}(K)}(\Delta x)^{2k+4} \leqslant C\|u\|_{W^{k+1,\infty}(\Omega)}(\Delta x)^{2k+2}.$$

Taking square roots on both sides finishes the proof. □

We verify in Appendix A that the two spaces (3.7) and (3.8) satisfy the condition (3.9) in Proposition 3.3.

We now show numerical results in Tables 3.3 and 3.4 for the approximations to $u(x,y) = e^{x-2y}$ and to $u(x,y) = \sin(x) + \sin(y)$ respectively. From the results in these two tables, we can see that we obtain the optimal order $k+1$ of the approximation rate when using the approximation spaces $E^k(\alpha,\beta)$ and $T^k(\alpha,\beta)$ $(k = 1, 2)$, to approximate general functions. It can again be seen that the approximation results are comparable to those obtained by the usual piecewise polynomial approximation for general functions. Not surprisingly, we obtain exact solutions when approximating specific functions tailored to the specific approximation spaces.

Table 3.3
L^2- and L^∞-errors of approximation to e^{x-2y} $(0 \leqslant x, y \leqslant 1)$. Uniform mesh with $N_x \times N_y$ cells

$N_x \times N_y$	Exponential space $E^1(1,-2)$				Trigonometric space $T^1(1,1)$			
	L^2-error		L^∞-error		L^2-error	Order	L^∞-error	Order
5 × 5	4.65E−15		2.49E−14		7.95E−03		6.65E−02	
10 × 10	5.57E−15		4.51E−14		2.00E−03	1.99	1.86E−02	1.84
20 × 20	4.66E−14		2.46E−13		5.01E−04	2.00	4.93E−03	1.92
40 × 40	1.43E−13		8.90E−13		1.25E−04	2.00	1.27E−03	1.96
80 × 80	3.80E−13		2.41E−12		3.13E−05	2.00	3.22E−04	1.98
160 × 160	2.27E−12		1.28E−11		7.84E−06	2.00	8.11E−05	1.99
	Exponential space $E^2(1,-2)$				Trigonometric space $T^2(1,1)$			
	L^2-error		L^∞-error		L^2-error	Order	L^∞-error	Order
5 × 5	8.30E−14		5.28E−13		4.05E−04		4.24E−03	
10 × 10	2.97E−13		1.88E−12		5.11E−05	2.99	6.00E−04	2.82
20 × 20	4.39E−12		2.52E−11		6.40E−06	3.00	7.98E−05	2.91
40 × 40	2.57E−11		1.62E−10		8.00E−07	3.00	1.03E−05	2.95
80 × 80	5.55E−11		3.42E−10		1.00E−07	3.00	1.31E−06	2.98
160 × 160	9.71E−11		9.46E−10		1.25E−08	3.00	1.67E−07	2.97
	Polynomial space P^1				Polynomial space P^2			
	L^2-error	Order	L^∞-error	Order	L^2-error	Order	L^∞-error	Order
5 × 5	7.94E−03		6.63E−02		3.81E−04		3.95E−03	
10 × 10	2.00E−03	1.99	1.86E−02	1.83	4.80E−05	2.99	5.61E−04	2.82
20 × 20	5.01E−04	2.00	4.93E−03	1.92	6.02E−06	3.00	7.47E−05	2.91
40 × 40	1.25E−04	2.00	1.27E−03	1.96	7.53E−07	3.00	9.65E−06	2.95
80 × 80	3.14E−05	1.99	3.22E−04	1.98	9.41E−08	3.00	1.23E−06	2.97
160 × 160	7.84E−06	2.00	8.11E−05	1.99	1.18E−08	3.00	1.54E−07	3.00

Table 3.4
L^2- and L^∞-errors of approximation to $\sin x + \sin y$ ($0 \leqslant x, y \leqslant \pi$). Uniform mesh with $N_x \times N_y$ cells

$N_x \times N_y$	Exponential space $E^1(1, 1)$				Trigonometric space $T^1(1, 1)$			
	L^2-error	Order	L^∞-error	Order	L^2-error	Order	L^∞-error	Order
5 × 5	1.94E−01		3.42E−01		4.58E−02		5.24E−02	
10 × 10	4.99E−02	1.96	8.97E−02	1.93	1.15E−02	1.99	1.30E−02	2.01
20 × 20	1.26E−02	1.99	2.25E−02	2.00	2.89E−03	1.99	3.30E−03	1.98
40 × 40	3.15E−03	2.00	5.61E−03	2.00	7.22E−04	2.00	8.26E−04	2.00
80 × 80	7.88E−04	2.00	1.40E−03	2.00	1.81E−04	2.00	2.07E−04	2.00
160 × 160	1.97E−04	2.00	3.49E−04	2.00	4.51E−05	2.00	5.17E−05	2.00
	Exponential space $E^2(1,1)$				Trigonometric space $T^2(1,1)$			
	L^2-error	Order	L^∞-error	Order	L^2-error		L^∞-error	
5 × 5	2.06E−02		5.00E−02		3.13E−13		3.52E−13	
10 × 10	2.65E−03	2.96	6.55E−03	2.93	3.97E−14		3.97E−14	
20 × 20	3.33E−04	2.99	8.30E−04	2.98	5.88E−14		6.44E−14	
40 × 40	4.17E−05	3.00	1.04E−04	3.00	5.18E−13		7.00E−13	
80 × 80	5.21E−06	3.00	1.30E−05	3.00	2.46E−12		2.73E−12	
160 × 160	6.52E−07	3.00	1.63E−06	3.00	4.00E−12		4.66E−12	
	Polynomial space P^1				Polynomial space P^2			
	L^2-error	Order	L^∞-error	Order	L^2-error	Order	L^∞-error	Order
5 × 5	4.60E−02		5.25E−02		2.44E−03		2.48E−03	
10 × 10	1.15E−02	2.00	1.31E−02	2.00	3.06E−04	3.00	3.21E−04	2.95
20 × 20	2.89E−03	1.99	3.30E−03	1.99	3.83E−05	3.00	4.05E−05	2.99
40 × 40	7.22E−04	2.00	8.26E−04	2.00	4.79E−06	3.00	5.07E−06	3.00
80 × 80	1.81E−04	2.00	2.07E−04	2.00	5.99E−07	3.00	6.34E−07	3.00
160 × 160	4.51E−05	2.00	5.17E−05	2.00	7.49E−08	3.00	7.93E−08	3.00

4. Discontinuous Galerkin method with time-varying approximation spaces

For the one-dimensional conservation law (2.1), we will use the space $E^k(\alpha)$ (3.1) or $T^k(\alpha)$ (3.3) in Section 3 as the finite element space. However, we would like to adjust the parameters α_j in these spaces for different time steps. The numerical scheme is set up as follows:

1. Decide the initial parameters α_j^0. We will calculate α_j^0 from the initial condition, using the same method for the choice of the initial parameters as for later parameter adjustment, to be described in detail in Section 4.1.
2. When the numerical solution $U^n \in V_h^n$ is known at time step n, where V_h^n is based on the choice of parameters α_j^n, use the DG or LDG method to obtain the preliminary numerical solution $\bar{U}^{n+1} \in V_h^n$ for the next time-step. This step is basically the same as that for DG or LDG method using the piecewise polynomial space. The only difference is that we use a non-polynomial finite element space V_h^n. For example, the DG method for the conservation law (2.1) is: Find $\bar{U}^{n+1}$ in the non-polynomial finite element space V_h^n such that

$$\int_{I_j} \left(\frac{\bar{U}^{n+1} - U^n}{\Delta t} \right) V \,\mathrm{d}x - \int_{I_j} f(U^n) V_x \,\mathrm{d}x + \widehat{f(U^n)}_{j+\frac{1}{2}} V^-_{j+\frac{1}{2}} - \widehat{f(U^n)}_{j-\frac{1}{2}} V^+_{j-\frac{1}{2}} = 0 \tag{4.1}$$

holds for all test function $V \in V_h^n$. The choice of the numerical fluxes $\widehat{f(U)}_{j+\frac{1}{2}}$ is the same as that for DG method based on the piecewise polynomial space. Notice that here we have used the forward Euler time stepping as an example to demonstrate the algorithm. In actual calculation we would use higher order TVD Runge–Kutta methods [20,12] which are convex combinations of the forward Euler operator.

3. Find the new parameters α_j^{n+1} which can better fit the preliminary numerical solution $\bar{U}^{n+1}$ at the new time step obtained from Step 2 above, hence the new finite element approximation space V_h^{n+1}. Again, this procedure will be described in detail in Section 4.1.
4. Use the L^2-projection to transfer the preliminary numerical solution $\bar{U}^{n+1} \in V_h^n$ to the numerical solution $U^{n+1} \in V_h^{n+1}$. Because the finite element space consists of discontinuous functions, the L^2-projection can be easily implemented locally. For example, when the cells do not change, we simply need to find the function $U^{n+1} \in V_h^{n+1}$ such that

$$\int_{I_j} (U^{n+1}(x) - \bar{U}^{n+1}(x))e_i(x)\mathrm{d}x = 0, \quad 0 \leqslant i \leqslant k, \tag{4.2}$$

   where $e_i(x)$, $0 \leqslant i \leqslant k$, form a basis of the new approximation space V_h^{n+1}. This involves only local, small linear system solvers.
5. Repeat Steps 2, 3, and 4 above until we reach the final time, or the steady state.

4.1. Methods to adjust the parameters

We will concentrate our attention on the exponential approximation space $E^k(\alpha)$ defined in (3.1), and describe our approach to find the parameters α_j in the basis functions of $E^k(\alpha)$ to fit the numerical solution. We determine the optimal α_j (denote it by β) in each cell by fitting the numerical solution $U(x)$ to the exponential function $c\mathrm{e}^{\beta(x-x_j)}$ on the cell I_j. In order to make the calculation easier, we choose to find the parameter β to minimize the L^2 difference between $\log|U(x)|$ and $\log|c\mathrm{e}^{\beta(x-x_j)}| = \log|c| + \beta(x - x_j)$ when c and β can both freely change, that is, we would like to find β such that

$$g(c,\beta) = \int_{I_j} (\log|U(x)| - \log|c| - \beta(x-x_j))^2 \mathrm{d}x$$

is minimized. Taking $\frac{\partial g}{\partial \beta}(c,\beta)$ and setting it to zero, we obtain

$$\int_{I_j} (\log|U(x)| - \log|c| - \beta(x-x_j))(x-x_j)\mathrm{d}x = 0. \tag{4.3}$$

This leads directly to

$$\beta = \frac{12}{(\Delta x_j)^3} \int_{I_j} (x-x_j)\log|U(x)|\mathrm{d}x. \tag{4.4}$$

If $U(x) \in V_h^n$ with the parameter α_j^n:

$$U(x) = \mathrm{e}^{\alpha_j^n(x-x_j)}(u_0 + u_1(x-x_j) + \cdots + u_k(x-x_j)^k), \tag{4.5}$$

then (4.4) becomes

$$\alpha_j^{n+1} = \beta = \alpha_j^n + \frac{12}{(\Delta x_j)^3} \int_{I_j} \log|u_0 + u_1(x-x_j) + \cdots + u_k(x-x_j)^k|\mathrm{d}x, \tag{4.6}$$

where the integral can be computed by numerical quadratures.

Generally, we can also attempt to fit the numerical solution $U(x)$ into an exponential-like function $\mathrm{e}^{\alpha_j(x-x_j)}\sum_{i=0}^{l}(x-x_j)^i$ for any $l < k$, but this will increase the difficulty of the calculation.

In Section 5.2 we will test the effectiveness of this method in identifying the parameter α_j. We have also tested a second method to determine the optimal α_j (denote it again by β) in each cell by fitting the numerical solution $U(x)$ to the exponential function $\mathrm{e}^{\beta(x-x_j)}(c_0 + c_1(x-x_j) + \cdots + c_k(x-x_j)^k)$ on the three cells I_{j-1}, I_j and I_{j+1}. The advantage of this second method is that it sometimes gives more accurate parameters than the first method based on our numerical experimental results. The disadvantage is that it requires more computational time because of its nonlinearity. We will not present numerical results associated with this second method to save space.

4.2. L^2-stability and an error estimate in one dimension

We present in this subsection the theoretical results of the L^2-stability for the general one-dimensional scalar nonlinear conservation laws (2.1) and an error estimate for the linear case

$$u_t + cu_x = 0, \tag{4.7}$$

where c is a constant.

Proposition 4.1 (L^2-stability). *Let u_h be the solution of the DG method (2.4) for the one-dimensional scalar nonlinear conservation law (2.1) based on a non-polynomial approximation space V_h. We have*

$$\|u_h(T)\|_{L^2}^2 + \Theta_T(u_h) \leqslant \|u_h(0)\|_{L^2}^2 \tag{4.8}$$

with $\Theta_T(u_h) \geqslant 0$. In particular, if the PDE (2.1) is the linear equation (4.7), then

$$\Theta_T(u_h) = |c| \int_0^{\mathrm{T}} \sum_{j=1}^N [u_h(t)]_{j+1/2}^2 \mathrm{d}t, \quad \text{where } [u_h]_{j+1/2} = u_h(x_{j+1/2}^+) - u_h(x_{j+1/2}^-).$$

The proof is identical to that for the polynomial approximation space case [14]. In fact, no special property of the approximation space V_h is used in the proof of the stability result in [14]. We do remark, however, that similar result can be proved for the fully discrete DG method only for special classes of time discretizations [14]. If the stability holds for fully discrete DG method based on a fixed approximation space V_h, it also holds for our modified DG method with the approximation space V_h^n changing with each time step, since we use the L^2 projection to transfer the preliminary numerical solution in the old space V_h^n to the new space V_h^{n+1}.

In order to prove an error estimate, we need the following approximation property of the finite element space V_h.

Lemma 4.2 (Approximation result). *Assume V_h is a space satisfying the condition (3.4) in Proposition 3.1 in each cell $I_j \in \Omega$. Let P_h be the L^2 projection operator into the space V_h. For any function $u(x) \in H^{k+1}(\Omega)$, there exists a constant C such that*

$$\left|P_h u(x_{j+1/2}^{\pm}) - u(x_{j+1/2})\right| \leqslant C(\Delta x)^{k+1/2} \|u\|_{H^{k+1}(I_j \cup I_{j+1})}.$$

Proof. We choose the same v_h as that in Proposition 3.1 and will use the same notations. We can decompose u as

$$u = v_h + (u - v_h).$$

Since $P_h(v_h) = v_h$, we have

$$|P_h u - u| \leqslant |P_h(u - v_h)| + |u - v_h|.$$

From Proposition 3.1, we know $|u - v_h| \leqslant C\|u\|_{H^{k+1}(I_j)}(\Delta x_j)^{k+1/2}$ for all $x \in I_j$. Now let

$$P_h(u - v_h) = \sum_{i=0}^k r_i v_i.$$

From the definition of the L^2-projection, we have

$$\int_{I_j} (P_h(u - v_h) - (u - v_h)) v_l \mathrm{d}x = 0 \quad 0 \leqslant l \leqslant k.$$

which can be written out as

$$\int_{I_j} \sum_{i=0}^k (r_i (\Delta x_j)^i) \frac{v_i(x)}{(\Delta x_j)^i} \frac{v_l(x)}{(\Delta x_j)^l} \mathrm{d}x = \int_{I_j} (u - v_h) \frac{v_l(x)}{(\Delta x_j)^l} \mathrm{d}x, \quad 0 \leqslant l \leqslant k$$

or equivalently

$$\int_{-1/2}^{1/2}\sum_{i=0}^{k}(r_i(\Delta x_j)^i)\frac{\tilde{v}_i(y)}{(\Delta x_j)^i}\frac{\tilde{v}_l(y)}{(\Delta x_j)^l}\,\mathrm{d}y=\int_{-1/2}^{1/2}(u-v_h)\frac{\tilde{v}_l(y)}{(\Delta x_j)^l}\,\mathrm{d}y,\quad 0\leqslant l\leqslant k,\tag{4.9}$$

where $y=\frac{x-x_j}{\Delta x_j}$ and $\tilde{v}_i(y)=v_i(x)$. (4.9) can be concisely written as

$$M_{\Delta x_j}R=h,$$

where $M_{\Delta x_j}=(m_{li})$ with $m_{li}=\int_{-1/2}^{1/2}\frac{\tilde{v}_i(y)}{(\Delta x_j)^i}\frac{\tilde{v}_l(y)}{(\Delta x_j)^l}\,\mathrm{d}y$, $R=\begin{pmatrix}r_0\\ r_1\Delta x_j\\ \dots\\ r_k(\Delta x_j)^k\end{pmatrix}$ and $h=\begin{pmatrix}h_0\\ h_1\\ \dots\\ h_k\end{pmatrix}$ with $h_l=\int_{-1/2}^{1/2}(u-v_h)\frac{\tilde{v}_l(y)}{(\Delta x_j)^l}\,\mathrm{d}y$. By (3.4), which implies the boundedness of $\frac{\tilde{v}_l(y)}{(\Delta x_j)^l}$, and (3.5), we easily obtain

$$|h_l|\leqslant C\|u\|_{H^{k+1}(I_j)}(\Delta x_j)^{k+1/2},\quad 0\leqslant l\leqslant k,$$

hence $\|h\|_\infty\leqslant C\|u\|_{H^{k+1}(I_j)}(\Delta x_j)^{k+1/2}$. Again from (3.4), we observe

$$\left|\frac{\tilde{v}_i(y)}{(\Delta x_j)^i}-a_{ii}y^i\right|\leqslant C\Delta x_j,$$

therefore the entries of the matrix $M_{\Delta x_j}$ can be written as

$$m_{li}=\int_{-1/2}^{1/2}a_{ii}y^i a_{ll}y^l\mathrm{d}y+\int_{-1/2}^{1/2}\left(\frac{v_i(y)}{(\Delta x_j)^i}\frac{v_l(y)}{(\Delta x_j)^l}-a_{ii}y^i a_{ll}y^l\right)\mathrm{d}y=f_{li}+g_{li}(\Delta x_j),$$

where f_{li} is independent of Δx_j and $|g_{li}(\Delta x_j)|\leqslant C\Delta x_j$. That is, we have

$$M_{\Delta x_j}=F+G(\Delta x_j),$$

where F is independent of Δx_j and is invertible (as it is the mass matrix of a local basis $\{a_{00}, a_{11}y, a_{22}y^2,\dots,a_{kk}y^k\}$ with $a_{ii}\neq 0$), and $\|G(\Delta x_j)\|_\infty\leqslant C\Delta x_j$. $M_{\Delta x_j}$ itself is also always invertible for any Δx_j, being the mass matrix of a local basis of V_h. Now, considering again (3.4) and the implied boundedness of $\frac{v_l(x)}{(\Delta x_j)^l}$, we have

$$\begin{aligned}\|P_h(u-v_h)\|_{L^\infty(I_j)}&\leqslant C\|R\|_\infty=C\|M_{\Delta x_j}^{-1}h\|_\infty\leqslant C\|M_{\Delta x_j}^{-1}\|_\infty\|h\|_\infty\leqslant C(\Delta x_j)^{k+1/2}\|u\|_{H^{k+1}(I_j)}\|(F+G(\Delta x_j))^{-1}\|_\infty\\ &\leqslant C(\Delta x_j)^{k+1/2}\|u\|_{H^{k+1}(I_j)}\|F^{-1}\|_\infty\|(I+F^{-1}G(\Delta x_j))^{-1}\|_\infty\\ &\leqslant C(\Delta x_j)^{k+1/2}\|u\|_{H^{k+1}(I_j)}\|F^{-1}\|_\infty\frac{1}{1-\|F^{-1}G(\Delta x_j))\|_\infty}\\ &\leqslant C(\Delta x_j)^{k+1/2}\|u\|_{H^{k+1}(I_j)}\|F^{-1}\|_\infty\frac{1}{1-\|F^{-1}\|_\infty C\Delta x_j}\leqslant C(\Delta x_j)^{k+1/2}\|u\|_{H^{k+1}(I_j)}.\end{aligned}$$

When Δx_j is suitably small, we now have

$$\|P_h u-u\|_{L^\infty(I_j)}\leqslant\|P_h(u-v_h)\|_{L^\infty(I_j)}+\|u-v_h\|_{L^\infty(I_j)}\leqslant C(\Delta x_j)^{k+1/2}\|u\|_{H^{k+1}(I_j)}.$$

Finally, we obtain

$$\begin{aligned}|P_h u(x_{j+1/2}^\pm)-u(x_{j+1/2})|&\leqslant\|P_h u-u\|_{L^\infty(I_j)}+\|P_h u-u\|_{L^\infty(I_{j+1})}\\ &\leqslant C(\Delta x_j)^{k+1/2}\|u\|_{H^{k+1}(I_j)}+C(\Delta x_{j+1})^{k+1/2}\|u\|_{H^{k+1}(I_{j+1})}\leqslant C(\Delta x)^{k+1/2}\|u\|_{H^{k+1}(I_j\cup I_{j+1})}.\end{aligned}$$

This finishes the proof. □

We are now ready to prove the following error estimate.

Proposition 4.3 (Error estimate). *Let u be the smooth exact solution of (4.7), and u_h be the numerical solution (2.4) by the DG method using a local space V_h satisfying the condition (3.4) and an upwind numerical flux $\widehat{f(U)}_{j+\frac{1}{2}} = \frac{c+|c|}{2}U^-_{j+\frac{1}{2}} + \frac{c-|c|}{2}U^+_{j+\frac{1}{2}}$. Then we have*

$$\|u-u_h\|_{L^2} \leqslant C\|u\|_{H^{k+1}}(\Delta x)^{k+1/2}. \tag{4.10}$$

Proof. Most of the proof of this proposition is the same as that for the standard piecewise polynomial space [10]. We denote $\bar{e} = P_h u - u_h$ and $\varepsilon = u - P_h u$, where $P_h u$ is the standard L^2 projection of u into the approximation space V_h, and take the test function $V = \bar{e}$ in (2.4) to obtain, after straightforward algebraic manipulations similar to the proof of the cell entropy inequalities [14]:

$$\frac{1}{2}\frac{\mathrm{d}}{\mathrm{d}t}\int(\bar{e})^2\mathrm{d}x + \frac{|c|}{4}\sum_j[\bar{e}^+_{j-1/2} - \bar{e}^-_{j-1/2}]^2 \leqslant \frac{c+|c|}{2}\sum_j(\varepsilon^-_{j-1/2})^2 + \frac{|c|-c}{2}\sum_j(\varepsilon^+_{j-1/2})^2.$$

From the result of Lemma 4.2, we have

$$\frac{1}{2}\frac{\mathrm{d}}{\mathrm{d}t}\int(\bar{e})^2\mathrm{d}x + \frac{|c|}{4}\sum_j[\bar{e}^+_{j-1/2} - \bar{e}^-_{j-1/2}]^2 \leqslant C\|u\|^2_{H^{k+1}}(\Delta x)^{2k+1}.$$

This clearly implies

$$\|\bar{e}(.,T)\|_{L^2} \leqslant C\|u\|_{H^{k+1}}(\Delta x)^{k+1/2}.$$

Combing with (3.6) of Proposition 3.2, we obtain

$$\|u-u_h\|_{L^2} \leqslant \|\varepsilon\|_{L^2} + \|\bar{e}\|_{L^2} \leqslant C\|u\|_{H^{k+1}}(\Delta x)^{k+1} + C\|u\|_{H^{k+1}}(\Delta x)^{k+1/2} \leqslant C\|u\|_{H^{k+1}}(\Delta x)^{k+1/2}.$$

This finishes the proof. □

The result of this proposition can be easily generalized to any one-dimensional linear systems.

Finally, we prove that the transfer of information from one approximation space to another does not destroy accuracy.

Proposition 4.4 (Error estimate for the projection). *If u is the smooth exact solution of (4.7), u_h is the numerical solution (2.4) by the DG method with V_h as in the previous proposition, and P_h is the L^2-projection into a new approximation space $\bar{V}_h$ satisfying the same condition (3.4), then we have*

$$\|u-P_h u_h\|_{L^2} \leqslant C\|u\|_{H^{k+1}}(\Delta x)^{k+1/2}. \tag{4.11}$$

Proof. First

$$\|u-P_h u_h\|_{L^2} \leqslant \|u-P_h u\|_{L^2} + \|P_h u - P_h u_h\|_{L^2} \leqslant \|u-P_h u\|_{L^2} + \|u-u_h\|_{L^2},$$

where the last inequality is because P_h is an L^2-projection hence it does not increase the L^2-norm. By Proposition 4.3, we have

$$\|u-u_h\|_{L^2} \leqslant C\|u\|_{H^{k+1}}(\Delta x)^{k+1/2}.$$

On the other hand, by (3.6), we have

$$\|u-P_h u\|_{L^2} \leqslant C\|u\|_{H^{k+1}}(\Delta x)^{k+1}.$$

Combining the two inequalities finishes the proof. □

4.3. Multi-dimensional case

The DG method with time-varying approximation spaces can be easily generalized from one-dimensional to multi-dimensional cases. We will comment on the numerical performance of the algorithm in Section 5. As to the stability and error estimate, applying the results in Section 3.2 and using similar proofs as in the one-dimensional case, the results in the previous subsection can be generalized to the multi-dimensional case. We will state the results for the two-dimensional case and will omit most of the proofs when they are similar to that in the one-dimensional case.

For a general scalar two-dimensional conservation law:

$$u_t + f(u)_x + g(u)_y = 0, \tag{4.12}$$

and its linear case:

$$u_t + c_x u_x + c_y u_y = 0, \tag{4.13}$$

we have the following stability result.

Proposition 4.5 (L^2-stability). *Let u_h be the solution of the DG method for the two-dimensional scalar nonlinear conservation law (4.12) based on a non-polynomial approximation space V_h. We have*

$$\|u_h(T)\|_{L^2}^2 + \Theta_T(u_h) \leqslant \|u_0\|_{L^2}^2 \tag{4.14}$$

with $\Theta_T(u_h) \geqslant 0$. In particular, if the PDE (4.12) is the linear equation (4.13), then

$$\Theta_T(u_h) = \int_0^{\mathrm{T}} \sum_{e\in\mathbb{E}_{\Delta x}} |c\cdot\mathbf{n}_{e,K}| \int_e [u_h(x,t)]^2 \,\mathrm{d}\Gamma(x)$$

with $c = (c_x, c_y)$, $\mathbb{E}_{\Delta x}$ is the collection of all edges, $\mathbf{n}_{e,\,K}$ is the outward unit normal along the edge e for the element K, and $[\,u_h] = u^{\mathrm{ext}(K)} - u^{\mathrm{int}(K)}$ denotes the jump of u along the edge.

As in the one-dimensional case, we would need the following approximation property of the finite element space V_h.

Lemma 4.6 (Approximation result). *Assume V_h is a space satisfying the condition (3.9) in Proposition 3.3 in each cell $K \in \Omega$. Let P_h be the L^2 projection operator into the space V_h. For any function $u(x) \in W^{k+1,\infty}(\Omega)$, there exists a constant C such that*

$$\|P_h u(x) - u(x)\|_{L^\infty(K)} \leqslant C\|u\|_{W^{k+1,\infty}(K)}(\Delta x)^{k+1}.$$

Proof. We choose the same v_h as that in Proposition 3.3 and will use the same notations. We can decompose u as

$$u = v_h + (u - v_h).$$

Since $P_h(v_h) = v_h$, we have

$$|P_h u - u| \leqslant |P_h(u - v_h)| + |u - v_h|.$$

From Proposition 3.3, we know $|u - v_h| \leqslant C\|u\|_{W^{k+1,\infty}(K)}(\Delta x)^{k+1}$ for all $x \in K$. Now let

$$P_h(u - v_h) = \sum_{m,n\geqslant 0, m+n\leqslant k} r_{mn} v_{mn}.$$

From the definition of the L^2-projection, we have

$$\int\int_K (P_h(u - v_h) - (u - v_h)) v_{pq} \mathrm{d}x\,\mathrm{d}y = 0, \quad p \geqslant 0,\ q \geqslant 0,\ p+q \leqslant k.$$

which can be written out as, for all $p, q \geqslant 0$, $p+q \leqslant k$,

$$\int\int_K \sum_{m,n\geqslant 0, m+n\leqslant k} (r_{mn}(\Delta x)^{m+n}) \frac{v_{mn}(x,y)}{(\Delta x)^{m+n}} \frac{v_{pq}(x,y)}{(\Delta x)^{p+q}} \mathrm{d}x\,\mathrm{d}y = \int\int_K (u - v_h) \frac{v_{pq}(x,y)}{(\Delta x)^{p+q}} \mathrm{d}x\,\mathrm{d}y$$

or equivalently, for all $p, q \geqslant 0$, $p+q \leqslant k$,

$$\int\int_{K'} \sum_{m,n\geqslant 0,m+n\leqslant k} (r_{mn}(\Delta x)^{m+n}) \frac{\tilde{v}_{mn}(\tilde{x},\tilde{y})}{(\Delta x)^{m+n}} \frac{\tilde{v}_{pq}(\tilde{x},\tilde{y})}{(\Delta x)^{p+q}} \,\mathrm{d}\tilde{x}\,\mathrm{d}\tilde{y} = \int\int_{K'} \frac{(u-v_h)\tilde{v}_{pq}(\tilde{x},\tilde{y})}{(\Delta x)^{p+q}} \,\mathrm{d}\tilde{x}\,\mathrm{d}\tilde{y}, \tag{4.15}$$

where $\tilde{x} = \frac{x-x_K}{\Delta x}, \tilde{y} = \frac{y-y_K}{\Delta x}, \tilde{v}_{mn}(\tilde{x},\tilde{y}) = v_{mn}(x,y)$ and $K' = \{(\tilde{x},\tilde{y}) : \Delta x(\tilde{x},\tilde{y}) + (x_K, y_K) \in K\}$. (4.15) can be concisely written as

$$M_{\Delta x} : R = h,$$

where $M_{\Delta x} = (m_{pqmn})$ with $m_{pqmn} = \int\int_{K'} \frac{\tilde{v}_{mn}(\tilde{x},\tilde{y})}{(\Delta x)^{m+n}} \frac{\tilde{v}_{pq}(\tilde{x},\tilde{y})}{(\Delta x)^{p+q}} \,\mathrm{d}\tilde{x}\,\mathrm{d}\tilde{y}$, $R = (r_{mn}(\Delta x)^{m+n})$ and $h = (h_{pq})$ with $h_{pq} = \int\int_{K'} \frac{(u-v_h)\tilde{v}_{pq}(\tilde{x},\tilde{y})}{(\Delta x)^{p+q}} \,\mathrm{d}\tilde{x}\,\mathrm{d}\tilde{y}$. By (3.9), which implies the boundedness of $\frac{\tilde{v}_{pq}(\tilde{x},\tilde{y})}{(\Delta x)^{p+q}}$, and (3.10), we easily obtain

$$|h_{pq}| \leqslant C\|u\|_{W^{k+1,\infty}(K)}(\Delta x)^{k+1}, \quad p \geqslant 0, q \geqslant 0, p+q \leqslant k. \tag{4.16}$$

We define a norm of the matrix $X = (x_{ij})$ as

$$\|X\|^{\infty} = \max_{i,j} |x_{ij}|,$$

and the associated operator norm of fourth order tensors M as

$$\|M\|^{\infty} = \sup_{X\neq 0} \frac{\|M : X\|^{\infty}}{\|X\|^{\infty}}.$$

From (4.16), we have $\|h\|^{\infty} \leqslant C\|u\|_{W^{k+1,\infty}(K)}(\Delta x)^{k+1}$. Again from (3.9), we observe

$$\left| \frac{\tilde{v}_{pq}(\tilde{x},\tilde{y})}{(\Delta x)^{p+q}} - a_{pqpq}\tilde{x}^p\tilde{y}^q \right| \leqslant C\Delta x,$$

therefore the entries of the tensor $M_{\Delta x}$ can be written as

$$\begin{aligned} m_{pqmn} &= \int\int_{K'} a_{mnmn}\tilde{x}^m\tilde{y}^n a_{pqpq}\tilde{x}^p\tilde{y}^q \,\mathrm{d}\tilde{x}\,\mathrm{d}\tilde{y} + \int\int_{K'} \left(\frac{\tilde{v}_{mn}(\tilde{x},\tilde{y})}{(\Delta x)^{m+n}} \frac{\tilde{v}_{pq}(\tilde{x},\tilde{y})}{(\Delta x)^{p+q}} - a_{mnmn}\tilde{x}^m\tilde{y}^n a_{pqpq}\tilde{x}^p\tilde{y}^q \right) \mathrm{d}\tilde{x}\,\mathrm{d}\tilde{y} \\ &= f_{pqmn} + g_{pqmn}(\Delta x), \end{aligned}$$

where f_{pqmn} is independent of Δx and $|g_{pqmn}(\Delta x)| \leqslant C\Delta x$. That is, we have

$$M_{\Delta x} = F + G(\Delta x),$$

where F is independent of Δx and its "inverse" $\bar{F}$ exists, as it is the mass matrix of a local basis $\{a_{0000}, a_{1010}\tilde{x}, a_{0101}\tilde{y}, \ldots, a_{k0k0}\tilde{x}^k, \ldots, a_{0k0k}\tilde{y}^k\}$ with $a_{mnmn} \neq 0$ (again, see Appendix A for the discussion of the inverse of such tensors), and $\|G(\Delta x)\|^{\infty} \leqslant C\Delta x$. The "inverse" of $M_{\Delta x}$ itself always exists for any Δx, because $M_{\Delta x}$ is the mass matrix of a local basis of V_h.

We need to define a new operation between two fourth order tensors F and G:

$$(F \circ G) : X = F : (G : X) \quad \text{for any matrix } X.$$

Setting $I = (i_{pqmn}) = (\delta_{pm}\delta_{qn})$, we have $F \circ I = F = I \circ F$ for any fourth order tensor F, and $F \circ \bar{F} = \bar{F} \circ F = I$ if F has an "inverse" $\bar{F}$ (see Appendix A). We can also easily verify from definition that the "inverse" of $F \circ G$ is $\bar{G} \circ \bar{F}$, i.e. $\overline{F \circ G} = \bar{G} \circ \bar{F}$. Now, considering again (3.9) and the implied boundedness of $\frac{v_{pq}(x,y)}{(\Delta x)^{p+q}}$, we have

$$\begin{aligned} \|P_h(u-v_h)\|_{L^{\infty}(K)} &\leqslant C\|R\|^{\infty} = C\|\bar{M}_{\Delta x} : h\|^{\infty} \leqslant C\|\bar{M}_{\Delta x}\|^{\infty}\|h\|^{\infty} \leqslant C(\Delta x)^{k+1}\|u\|_{W^{k+1,\infty}(K)}\|\overline{F + G(\Delta x)}\|^{\infty} \\ &\leqslant C(\Delta x)^{k+1}\|u\|_{W^{k+1,\infty}(K)}\|\bar{F}\|^{\infty}\|\overline{I + \bar{F} \circ G(\Delta x)}\|^{\infty} \\ &\leqslant C(\Delta x)^{k+1}\|u\|_{W^{k+1,\infty}(K)}\|\bar{F}\|^{\infty} \frac{1}{1 - \|\bar{F} \circ G(\Delta x))\|^{\infty}} \\ &\leqslant C(\Delta x)^{k+1}\|u\|_{W^{k+1,\infty}(K)}\|\bar{F}\|^{\infty} \frac{1}{1 - \|\bar{F}\|^{\infty}C\Delta x} \leqslant C(\Delta x)^{k+1}\|u\|_{W^{k+1,\infty}(K)}. \end{aligned}$$

When Δx is suitably small, we now have

$$\|P_h u - u\|_{L^\infty(K)} \leqslant \|P_h(u - v_h)\|_{L^\infty(K)} + \|u - v_h\|_{L^\infty(K)} \leqslant C\|u\|_{W^{k+1,\infty}(K)}(\Delta x)^{k+1}.$$

This finishes the proof. □

With this approximation result we can prove the following error estimate.

Proposition 4.7 (Error estimate). *Let u be the smooth exact solution of (4.13), and u_h be the numerical solution by the DG method using a local space V_h satisfying the condition (3.9) and an upwind numerical flux. Then we have*

$$\|u - u_h\|_{L^2} \leqslant C\|u\|_{W^{k+1,\infty}}(\Delta x)^{k+1/2}. \tag{4.17}$$

The result of this proposition can be easily generalized to any two-dimensional symmetric linear systems.

Finally, we have the same error estimate for the transfer of information between different approximation spaces.

Proposition 4.8 (Error estimate for projection). *If u is the smooth exact solution of (4.13), u_h is the numerical solution by the DG method with V_h as in the previous proposition, and P_h is the L^2-projection into a new approximation space $\bar{V}_h$ satisfying the same condition (3.9), then we have*

$$\|u - P_h u_h\|_{L^2} \leqslant C\|u\|_{W^{k+1,\infty}}(\Delta x)^{k+1/2}. \tag{4.18}$$

5. Numerical examples

In this section, we present selected numerical experimental results to demonstrate the performance of our DG method based on non-polynomial approximation spaces, including the performance of the methods to determine the parameters in the local spaces. We concentrate our attention mostly on the exponential spaces, and use uniform Cartesian meshes to simplify the implementation, although the method can be easily applied on general triangulations. Time discretization is via the TVD Runge–Kutta time discretization [20,12] of order comparable to the spatial accuracy.

5.1. Shock capturing

We compute the one-dimensional Burgers equation to demonstrate the accuracy and shock capturing capability of the DG method based on non-polynomial approximation spaces for nonlinear conservation laws.

Example 5.1. One-dimensional Burgers equation:

$$u_t + \left(\frac{u^2}{2}\right)_x = 0 \tag{5.1}$$

with the initial condition

Table 5.1
Example 5.1. Errors of Burgers equation at time $T = 0.05$ with the TVB limiter ($M = 40$). N uniform cells

N	Polynomial space P^2				Exponential space $E^2(1)$			
	L^2-error	Order	L^∞-error	Order	L^2-error	Order	L^∞-error	Order
10	1.48E−03		4.82E−03		1.50E−03		5.46E−03	
20	2.32E−04	2.67	9.32E−04	2.37	2.46E−04	2.61	1.02E−03	2.42
40	3.45E−05	2.75	1.49E−04	2.65	3.95E−05	2.64	1.61E−04	2.66
80	5.11E−06	2.76	2.16E−05	2.79	5.82E−06	2.76	2.58E−05	2.64
160	7.05E−07	2.86	3.02E−06	2.84	8.18E−07	2.83	3.93E−06	2.71
320	9.21E−08	2.94	4.02E−07	2.91	1.08E−07	2.92	5.28E−07	2.90

$$u(x,0) = \frac{1}{2} + \sin 2\pi x \tag{5.2}$$

and a periodic boundary condition.

The exact solution is smooth at $T = 0.05$ and has a well developed shock at $T = 0.5$. In Table 5.1, we can see, whereas when the TVB generalized slope limiter is used with a properly chosen constant $M = 40$ (see [7] for the definition of this limiter), the DG method is uniformly high order using both P^2 and $E^2(1)$ spaces. The solution at $T = 0.5$ is plotted in Fig. 5.1 and it can be seen that the numerical solutions under the two approximation spaces are almost the same, i.e. the DG method based on the exponential space is able to capture the shock well.

5.2. One-dimensional problems: parameter adjustment in the local spaces

Example 5.2. We solve the PDE

$$u_t + u_x = 2u, \quad 0 \leqslant x \leqslant \pi, \tag{5.3}$$

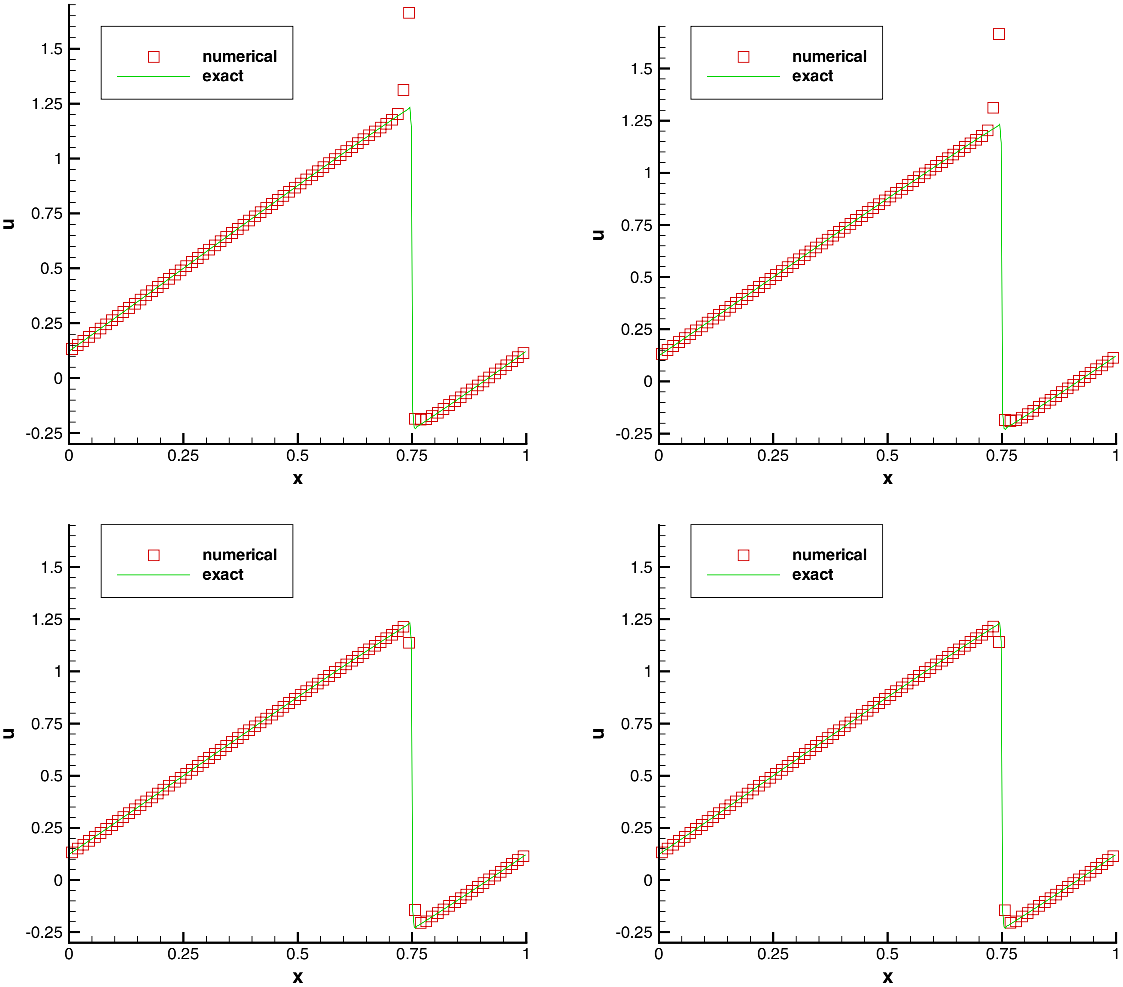

Fig. 5.1. Example 5.1. Solutions of Burgers equation at time $T = 0.5$. Uniform mesh with 80 cells. Top: no limiters; Bottom: with the TVB limiter, $M = 40$. Left: P^2 approximation space; Right: $E^2(1)$ approximation space. Solid line: the exact solution; Symbols: numerical solutions (one point per cell).

with a constant initial condition: $u(x,0) = 1$ and the boundary condition: $u(0,t) = 1$. The exact steady state solution is

$$u(x) = \mathrm{e}^{2x}. \tag{5.4}$$

This example has the steady state solution in the approximation space $E^k(\alpha)$, if the method to determine the local parameter α_j is able to catch the correct α value. Notice that the constant initial condition is far from the final steady state, so the correct identification of α is possible only at a later stage of time evolution.

The numerical solution and parameter adjustment results are shown in Tables 5.2 and 5.3. From Table 5.2, we can see that by using the exponential space $E^k(\alpha)$, we obtain basically the exact solution modulo round off errors, which are far better than the solutions obtained by using the standard P^k space. From Table 5.3, we find the parameters are adjusted gradually as time grows and eventually reach the values fitting the steady state solution almost exactly.

Table 5.2
Example 5.2. L^2- and L^∞-errors at steady state under two different approximation spaces. N uniform cells

N	Polynomial space P^1				Exponential space $E^1(\alpha)$	
	L^2-error	Order	L^∞-error	Order	L^2-error	L^∞-error
10	9.40E−00		3.20E+01		4.41E−11	1.56E−10
20	1.83E−00	2.36	8.09E−00	1.98	3.61E−12	7.09E−12
40	4.21E−01	2.12	1.99E−00	2.02	1.30E−11	4.57E−11
80	1.02E−01	2.05	4.88E−01	2.03	1.19E−10	4.50E−10
160	2.53E−02	2.01	1.21E−01	2.01	8.90E−10	4.97E−09
	Polynomial space P^2				Exponential space $E^2(\alpha)$	
	L^2-error	Order	L^∞-error	Order	L^2-error	L^∞-error
10	3.38E−01		1.30E−00		3.33E−11	6.91E−11
20	4.16E−02	3.02	1.79E−01	2.86	8.17E−11	1.75E−10
40	5.14E−03	3.02	2.35E−02	2.93	6.23E−11	3.53E−10
80	6.38E−04	3.01	3.01E−03	2.96	1.26E−11	4.12E−11
160	7.94E−05	3.01	3.82E−04	2.98	5.29E−11	4.71E−10

Table 5.3
Example 5.2. The parameters α_j for each cell at time T. $N = 10$ uniform cells

Time T	Parameters found at time T for each cell I_j									
	1	2	3	4	5	6	7	8	9	10
Exponential space $E^1(\alpha)$										
0.0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
1.0	1.34	6.08	6.37	4.18	2.31	1.28	0.90	0.71	0.58	0.49
2.0	2.01	2.02	1.28	2.10	6.61	5.56	3.94	2.48	1.46	0.95
3.0	2.00	2.00	2.01	2.09	1.81	0.57	3.99	6.28	5.19	3.84
4.0	2.00	2.00	2.00	2.00	1.99	2.06	2.13	1.47	0.30	4.87
5.0	2.00	2.00	2.00	2.00	2.00	2.00	1.99	2.00	2.14	2.09
6.0	2.00	2.00	2.00	2.00	2.00	2.00	2.00	2.00	2.00	1.97
7.0	2.00	2.00	2.00	2.00	2.00	2.00	2.00	2.00	2.00	2.00
Exponential space $E^2(\alpha)$										
0.0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
1.0	2.07	0.11	11.0	7.94	2.42	1.39	0.95	0.73	0.60	0.50
2.0	2.00	2.00	1.94	2.27	−0.57	10.3	8.39	3.12	1.44	1.01
3.0	2.00	2.00	2.00	2.00	2.02	1.90	2.46	−1.22	9.70	8.82
4.0	2.00	2.00	2.00	2.00	2.00	2.00	2.00	2.03	1.85	2.56
5.0	2.00	2.00	2.00	2.00	2.00	2.00	2.00	2.00	2.00	2.00

Example 5.3. We solve the PDE

$$u_t + u_x = 2xu, \quad 0 \leqslant x \leqslant 1, \tag{5.5}$$

with a constant initial condition: $u(x, 0) = 1$ and the boundary condition: $u(0, t) = 1$. The exact steady state solution is

$$u(x) = \mathrm{e}^{x^2}. \tag{5.6}$$

This example has the steady state solution which is not in the approximation space $E^k(\alpha)$ for any α, but may be approximated better locally by $E^k(\alpha)$ with a suitably chosen α_j than by the standard polynomial space P^k. Notice that the constant initial condition is again far from the final steady state, so as before the correct identification of α is possible only at a later stage of time evolution.

The numerical solution and parameter adjustment results are shown in Tables 5.4 and 5.5. From Table 5.4, we can see that the exponential space $E^k(\alpha)$ and the standard polynomial space P^k both have the same convergence rate in L^2 and L^∞ norms. However, the errors of exponential space are much smaller than those of the polynomial space, indicating that we have obtained good parameters in the exponential space so that better

Table 5.4
Example 5.3. L^2- and L^∞-errors at steady state under two different approximation spaces. N uniform cells

N	Polynomial space P^1				Exponential space $E^1(\alpha)$			
	L^2-error	Order	L^∞-error	Order	L^2-error	Order	L^∞-error	Order
10	4.16E−03		2.09E−02		1.92E−03		7.67E−03	
20	1.03E−03	2.01	5.55E−03	1.91	4.74E−04	2.02	1.95E−03	1.98
40	2.55E−04	2.01	1.43E−03	1.96	1.18E−04	2.01	4.90E−04	1.99
80	6.34E−05	2.01	3.63E−04	1.98	2.93E−05	2.01	1.23E−04	1.99
160	1.58E−05	2.00	9.15E−05	1.99	7.32E−06	2.00	3.07E−05	2.00
	Polynomial space P^2				Exponential space $E^2(\alpha)$			
	L^2-error	Order	L^∞-error	Order	L^2-error	Order	L^∞-error	Order
10	9.93E−05		5.55E−04		2.45E−06		1.02E−05	
20	1.23E−05	3.01	7.50E−05	2.89	2.60E−07	3.23	1.19E−06	3.10
40	1.53E−06	3.01	9.75E−06	2.94	1.84E−08	3.82	1.43E−07	3.05
80	1.91E−07	3.00	1.24E−06	2.98	2.04E−09	3.17	1.73E−08	3.04
160	2.39E−08	3.00	1.57E−07	2.98	2.41E−10	3.08	2.14E−09	3.02

Table 5.5
Example 5.3. The parameters α_j for each cell at time T. $N = 10$ uniform cells

Time T	Parameters found at time T for each cell I_j									
	1	2	3	4	5	6	7	8	9	10
Exponential space $E^1(\alpha)$										
0.0	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
0.1	0.65	1.63	1.53	1.34	1.23	1.13	1.05	0.98	0.92	0.87
1.0	0.13	0.33	0.53	0.73	0.94	1.14	1.32	1.49	1.86	2.58
2.0	0.13	0.33	0.53	0.73	0.94	1.14	1.33	1.54	1.74	1.94
3.0	0.13	0.33	0.53	0.73	0.94	1.14	1.34	1.54	1.74	1.94
4.0	0.13	0.33	0.53	0.73	0.94	1.14	1.34	1.54	1.74	1.94
Exponential space $E^2(\alpha)$										
0.0	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
0.1	6.62	3.06	1.90	1.38	1.24	1.15	1.07	1.00	0.95	0.90
1.0	0.10	0.30	0.50	0.70	0.90	1.10	1.32	1.49	0.93	8.81
2.0	0.10	0.30	0.50	0.70	0.90	1.10	1.30	1.50	1.70	1.90
3.0	0.10	0.30	0.50	0.70	0.90	1.10	1.30	1.50	1.70	1.90

approximations are obtained to the exact solution. Table 5.5 shows that the parameters are adjusted gradually as time grows and eventually reach values fitting the steady state solution well (since $e^{x^2} = e^{x_j^2} e^{2x_j(x-x_j)} e^{(x-x_j)^2}$, where $e^{x_j^2}$ is a constant and $e^{(x-x_j)^2} \approx 1$, the best fitting parameter is $2x_j$ in cell I_j).

Example 5.4. We solve the boundary layer problem:

$$u_t + u_x = \varepsilon u_{xx}, \quad 0 \leqslant x \leqslant 0.5, \tag{5.7}$$

where $\varepsilon = 0.01$, with a linear initial condition: $u(x, 0) = 2x$ and the boundary conditions $u(0, t) = 0$ and $u(0.5, t) = 1$. The exact steady state solution is

$$u(x) = \frac{e^{x/\varepsilon} - 1}{e^{1/2\varepsilon} - 1}. \tag{5.8}$$

This example has the steady state solution which has a sharp boundary layer at the right boundary, which can be approximated by the space $E^k(\alpha)$ with suitably chosen α much better than by the standard polynomial space P^k. Again, the initial condition is far from the final steady state, hence the correct identification of α is possible only at a later stage of time evolution.

The numerical solution and parameter adjustment results are given in Tables 5.6 and 5.7. In Table 5.6, we can see that by using the exponential space $E^k(\alpha)$, we obtain basically the exact solution modulo round off errors, which are far better than the solutions obtained by using the standard P^k space. From Table 5.7, we find the parameters are adjusted gradually as time grows and eventually reach the value $1/\varepsilon = 100$ fitting the steady state solution almost exactly.

Example 5.5. We solve the PDE

$$u_t = (p(x)u)_x + q(x)u. \tag{5.9}$$

1. $p(x) = x^2$, $q(x) = -x^2 e^x - 2x + 1$ and $1 \leqslant x \leqslant 2$ with the constant initial condition: $u(x, 0) = 2$ and the boundary condition: $u(2, t) = 2$. The exact steady state solution is $u(x) = 2e^{e^x + 1/x - e^2 - 1/2}$. Table 5.8 gives the results.
2. $p(x) = -x$, $q(x) = x \cos x + x^2 + 1$ and $1 \leqslant x \leqslant 2$ with the constant initial condition: $u(x, 0) = 2$ and the boundary condition: $u(1, t) = 2$. The exact steady state solution is $u(x) = 2e^{\sin x + x^2/2 - \sin 1 - 1/2}$. The results are given in Table 5.9.

From Tables 5.8 and 5.9, we can clearly observe that the exponential space $E^k(\alpha)$ and the standard polynomial space P^k both have the same convergence rate in L^2 and L^∞ norms. However, the errors of exponential space are much smaller than those of the polynomial space, indicating that we have obtained good parameters in the exponential space so that better approximations are obtained to the exact solution.

Table 5.6
Example 5.4. L^2- and L^∞-errors at steady state under two different approximation spaces. N uniform cells

N	Polynomial space P^1				Exponential space $E^1(\alpha)$	
	L^2-error	Order	L^∞-error	Order	L^2-error	L^∞-error
10	3.14E−02		3.55E−01		2.35E−14	1.68E−13
20	1.45E−02	1.11	2.12E−01	0.74	6.82E−14	5.22E−13
40	5.14E−03	1.50	1.03E−01	1.04	1.51E−13	1.45E−12
80	1.52E−03	1.76	3.73E−02	1.47	3.78E−13	2.92E−12
	Polynomial space P^2				Exponential space $E^2(\alpha)$	
	L^2-error	Order	L^∞-error	Order	L^2-error	L^∞-error
10	1.20E−02		1.24E−01		5.97E−14	4.51E−13
20	3.13E−03	2.27	4.56E−02	1.70	1.09E−14	7.61E−14
40	5.51E−04	2.71	1.10E−02	2.17	2.76E−14	2.15E−13
80	7.87E−05	2.91	1.98E−03	2.53	2.80E−15	3.73E−14

Table 5.7
Example 5.4. The parameters α_j for each cell at time T. $N = 10$ uniform cells

Time T	Parameters found at time T for each cell I_j									
	1	2	3	4	5	6	7	8	9	10
Exponential space $E^1(\alpha)$										
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
1.0	62.7	51.8	47.4	42.9	43.1	30.3	74.5	99.9	100.0	100.0
1.5	64.6	43.6	94.0	99.9	100.0	100.0	100.0	100.0	100.0	100.0
2.0	100.2	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0
3.0	100.2	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0
Exponential space $E^2(\alpha)$										
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
1.0	78.9	58.2	51.4	47.1	44.0	43.5	57.5	97.7	100.0	100.0
1.5	81.3	69.1	89.4	99.3	100.0	100.0	100.0	100.0	100.0	100.0
2.0	102.6	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0
3.0	102.6	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0

Table 5.8
Example 5.5. L^2- and L^∞-errors at steady state under two different approximation spaces. N uniform cells

N	Polynomial space P^1				Exponential space $E^1(\alpha)$			
	L^2-error	Order	L^∞-error	Order	L^2-error	Order	L^∞-error	Order
5	4.64E−02		2.32E−01		7.00E−03		2.73E−02	
10	1.43E−02	1.70	9.45E−02	1.30	2.07E−03	1.76	1.20E−02	1.19
20	3.94E−03	1.86	3.12E−02	1.60	5.58E−04	1.89	4.05E−03	1.57
40	1.03E−03	1.94	9.06E−03	1.78	1.45E−04	1.94	1.18E−03	1.78
80	2.62E−04	1.98	2.45E−03	1.89	3.68E−05	1.98	3.19E−04	1.89
	Polynomial space P^2				Exponential space $E^2(\alpha)$			
	L^2-error	Order	L^∞-error	Order	L^2-error	Order	L^∞-error	Order
5	6.57E−03		3.29E−02		1.09E−04		4.59E−04	
10	1.03E−03	2.68	7.00E−03	2.23	1.48E−05	2.88	9.18E−05	2.32
20	1.40E−04	2.87	1.17E−03	2.58	1.98E−06	2.90	1.52E−05	2.59
40	1.81E−05	2.95	1.70E−04	2.78	2.58E−07	2.94	2.22E−06	2.78
80	2.30E−06	2.98	2.30E−05	2.89	3.29E−08	2.97	3.01E−07	2.88

Table 5.9
Example 5.5. L^2- and L^∞-errors at steady state under two different approximation spaces. N uniform cells

N	Polynomial space P^1				Exponential space $E^1(\alpha)$			
	L^2-error	Order	L^∞-error	Order	L^2-error	Order	L^∞-error	Order
5	3.59E−02		1.45E−01		7.94E−04		3.50E−03	
10	8.47E−03	2.08	3.63E−02	2.00	1.75E−04	2.18	1.06E−03	1.72
20	2.07E−03	2.03	9.06E−03	2.00	4.16E−05	2.07	2.89E−04	1.87
40	5.11E−04	2.02	2.26E−03	2.00	1.02E−05	2.03	7.55E−05	1.94
80	1.27E−04	2.01	5.65E−04	2.00	2.52E−06	2.02	1.93E−05	1.97
	Polynomial space P^2				Exponential space $E^2(\alpha)$			
	L^2-error	Order	L^∞-error	Order	L^2-error	Order	L^∞-error	Order
5	9.18E−04		3.88E−03		5.83E−05		2.88E−04	
10	1.12E−04	3.03	5.15E−04	2.91	7.25E−06	3.01	4.18E−05	2.78
20	1.39E−05	3.02	6.66E−05	2.95	8.98E−07	3.01	5.59E−06	2.90
40	1.72E−06	3.01	8.47E−06	2.98	1.11E−07	3.01	7.21E−07	2.95
80	2.15E−07	3.00	1.07E−06	2.98	1.65E−08	2.75	1.05E−07	2.78

5.3. Helmholtz equation

Example 5.6. We consider the equation:

$$u_t = u_{xx} + \lambda u \tag{5.10}$$

and we are interested in the steady state solution of (5.10) which is the one-dimensional Helmholtz equation. When $\lambda > 0$, the exact solution can be written in the form of $c_1 \sin\sqrt{\lambda}x + c_2 \cos\sqrt{\lambda}x$. The trigonometric space $T^k(\sqrt{\lambda})$ given by (3.3) can be used in the LDG method to obtain almost exact numerical solution. When $\lambda < 0$, the exact solution can be written as $c_1 e^{\sqrt{-\lambda}x} + c_2 e^{-\sqrt{-\lambda}x}$. Usually, the exact solution is dominated by one term, either $c_1 e^{\sqrt{-\lambda}x}$ or $c_2 e^{-\sqrt{-\lambda}x}$. The time-varying exponential space $E^k(\alpha)$ can be applied in this case.

1. $\lambda = 4$ with a linear initial condition:

   $$u(x,0) = \frac{4}{\pi}x + 1, \quad 0 \leqslant x \leqslant \pi/4, \tag{5.11}$$

   and the boundary conditions $u(0, t) = 1$ and $u(\pi/4, t) = 2$. The exact steady state solution is

   $$u(x) = 2\sin 2x + \cos 2x. \tag{5.12}$$

   The numerical results are shown in Table 5.10. We can see that the exact solution (modulo round off errors) is obtained by the LDG method using the approximation space $T^2(2)$.
2. $\lambda = -9$ with a linear initial condition:

   $$u(x,0) = (e^{-3} + 2e^{3} - 3)x + 3, \quad 0 \leqslant x \leqslant 1, \tag{5.13}$$

   and the boundary conditions $u(0, t) = 3$ and $u(1, t) = e^{-3} + 2e^{3}$. The exact steady state solution is

   $$u(x) = e^{-3x} + 2e^{3x}. \tag{5.14}$$

   The numerical results are given in Table 5.11. We can see that the exponential space $E^k(\alpha)$ and the standard polynomial space P^k both have the same convergence rate in L^2 and L^∞ norms. However, the errors of exponential space are much smaller than those of the polynomial space, indicating that we have obtained good parameters in the exponential space so that better approximations are obtained to the exact solution.

5.4. Two-dimensional problems

The two-dimensional numerical procedure follows that in one dimension. We observe that the identification of the optimal parameters in our non-polynomial DG method is more time consuming than the one-dimensional case. This is due partly to the increased cost of the base algorithm in two dimensions, and partly to the fact that more parameters (e.g. two instead of one for the exponential spaces) must be identified. We have therefore made the following two modifications to speed up the computation. When using $E^k(\alpha, \beta)$ space to solve two-dimensional problems, we first solve the problems using DG method with polynomial P^k space to reach steady state and use this result as the initial condition. We then apply the parameter adjustment method, namely the method to fit the parameters described in Section 4.1 for the one-dimensional case, with apparent modification for the two-dimensional case, to evolve in the $E^k(\alpha, \beta)$ space with adjusted α and β towards steady states. In this stage we have not used the logarithm in the least square procedure. This makes

Table 5.10
Example 5.6. L^2- and L^∞-errors at steady state under two different approximation spaces. N uniform cells

N	Polynomial space P^2				Trigonometric space $T^2(2)$	
	L^2-error	Order	L^∞-error	Order	L^2-error	L^∞-error
5	2.50E−04		1.44E−03		1.66E−15	4.88E−15
10	2.51E−05	3.30	1.28E−04	3.02	1.38E−15	5.33E−15
20	2.72E−06	3.19	1.57E−05	3.03	4.14E−15	2.09E−14
40	3.15E−07	3.11	1.95E−06	3.01	1.66E−14	9.06E−14
80	3.78E−08	3.06	2.43E−07	3.00	2.94E−15	1.55E−14

Table 5.11
Example 5.6. L^2- and L^∞-errors under two different approximation spaces. N uniform cells

N	Polynomial space P^2				Exponential space $E^2(\alpha)$			
	L^2-error	Order	L^∞-error	Order	L^2-error	Order	L^∞-error	Order
5	2.98E−02		1.43E−01		1.98E−03		7.16E−03	
10	3.51E−03	3.09	2.21E−02	2.69	2.44E−04	3.02	9.31E−04	2.94
20	3.87E−04	3.18	3.07E−03	2.85	3.04E−05	3.00	1.16E−04	3.00
40	4.26E−05	3.18	4.06E−04	2.92	3.80E−06	3.00	1.44E−05	3.00
80	4.83E−06	3.14	5.22E−05	2.96	4.75E−07	3.00	1.79E−06	3.00

the least square procedure more costly for each step but it seems to allow us to reach the numerically optimal parameters much faster.

Example 5.7. We consider the following two-dimensional boundary layer problem:

$$u_t + u_x + u_y = \varepsilon_x u_{xx} + \varepsilon_y u_{yy}, \quad 0 \leqslant x, y \leqslant 1. \tag{5.15}$$

1. $\varepsilon_x = \varepsilon_y = 0.05$ with an initial condition

   $$u(x, y, 0) = 1 \tag{5.16}$$

   and the boundary conditions $u(x, 0, t) = \mathrm{e}^{20x-40}$, $u(x, 1, t) = \mathrm{e}^{20x-20}$, $u(0, y, t) = \mathrm{e}^{20y-40}$ and $u(1, y, t) = \mathrm{e}^{20y-20}$. The exact steady state solution is $u(x, y) = \mathrm{e}^{20(x+y-2)}$. The numerical results are given in Table 5.12.
2. $\varepsilon_x = 0.05$, $\varepsilon_y = 0.5$ with an initial condition

   $$u(x, y, 0) = 1 \tag{5.17}$$

   and the boundary conditions $u(x, 0, t) = \mathrm{e}^{20x-20}$, $u(x, 1, t) = \mathrm{e}^{20x-18}$, $u(0, y, t) = \mathrm{e}^{2y-20}$ and $u(1, y, t) = \mathrm{e}^{2y}$. The exact steady state solution is $u(x, y) = \mathrm{e}^{20x+2y-20}$. The numerical results are given in Table 5.13.

We can see clearly that, as in the one-dimensional case, by using the exponential space $E^k(\alpha, \beta)$, our method can identify automatically the correct values of α and β based on the numerical solution, which leads to basically the exact solution modulo round off errors, which are far better than the solutions obtained by using the standard P^k space.

Example 5.8. We consider the equation:

$$u_t = u_{xx} + u_{yy} + \lambda u, \quad 0 \leqslant x, y \leqslant 1 \tag{5.18}$$

and solve the steady state solution of (5.18) which is the two-dimensional Helmholtz equation.

Table 5.12
Example 5.7. L^2- and L^∞-errors at steady state under two different approximation spaces. $N_x \times N_y$ uniform cells

$N_x \times N_y$	Polynomial space P^1				Exponential space $E^1(\alpha, \beta)$	
	L^2-error	Order	L^∞-error	Order	L^2-error	L^∞-error
10 × 10	7.02E−03		3.60E−01		9.23E−11	4.42E−10
20 × 20	2.24E−03	1.65	1.69E−01	1.09	1.11E−11	7.39E−10
40 × 40	6.01E−04	1.90	6.00E−02	1.65	1.15E−12	9.38E−11
80 × 80	1.53E−04	2.97	1.81E−02	1.73	8.66E−12	3.10E−10
	Polynomial space P^2				Exponential space $E^2(\alpha, \beta)$	
	L^2-error	Order	L^∞-error	Order	L^2-error	L^∞-error
10 × 10	2.13E−03		1.25E−01		8.81E−13	4.42E−11
20 × 20	3.54E−04	2.59	3.22E−02	1.96	5.28E−13	3.41E−11
40 × 40	4.81E−05	2.90	5.99E−03	2.43	2.89E−13	2.40E−11
80 × 80	6.15E−06	2.97	9.21E−04	2.70	8.66E−12	3.10E−10

Table 5.13
Example 5.7. L^2- and L^∞-errors at steady state under two different approximation spaces. $N_x \times N_y$ uniform cells

$N_x \times N_y$	Polynomial space P^1				Exponential space $E^1(\alpha, \beta)$	
	L^2-error	Order	L^∞-error	Order	L^2-error	L^∞-error
10 × 10	7.15E−02		1.16E−00		1.44E−12	1.74E−10
20 × 20	2.08E−02	1.78	4.29E−01	1.44	8.63E−12	1.30E−10
40 × 40	5.44E−03	1.93	1.32E−01	1.70	3.07E−12	7.03E−10
80 × 80	1.38E−03	1.98	3.70E−02	1.83	8.87E−12	1.80E−10
	Polynomial space P^2				Exponential space $E^2(\alpha, \beta)$	
	L^2-error	Order	L^∞-error	Order	L^2-error	L^∞-error
10 × 10	1.22E−02		2.06E−01		1.51E−11	1.74E−10
20 × 20	1.81E−03	2.75	4.00E−02	2.34	1.82E−11	2.82E−10
40 × 40	2.37E−04	2.93	6.31E−03	2.66	2.87E−12	6.71E−11
80 × 80	3.00E−05	2.98	8.88E−04	2.83	1.65E−11	3.73E−10

Table 5.14
Example 5.8. L^2- and L^∞-errors at steady state under two different approximation spaces. $N_x \times N_y$ uniform cells

$N_x \times N_y$	Polynomial space P^1				Exponential space $E^1(\alpha, \beta)$			
	L^2-error	Order	L^∞-error	Order	L^2-error	Order	L^∞-error	Order
10 × 10	7.36E00		1.06E02		2.01E−01		1.28E00	
20 × 20	1.85E00	1.99	2.92E01	1.86	5.05E−02	1.99	3.39E−01	1.92
40 × 40	4.63E−01	2.00	7.68E00	1.93	1.26E−02	2.00	8.70E−02	1.96
80 × 80	1.16E−01	2.00	1.97E00	1.96	3.16E−03	2.00	2.21E−02	1.98
	Polynomial space P^2				Exponential space $E^2(\alpha, \beta)$			
	L^2-error	Order	L^∞-error	Order	L^2-error	Order	L^∞-error	Order
10 × 10	2.96E−01		5.37E00		9.11E−03		5.98E−02	
20 × 20	3.72E−02	2.99	7.47E−01	2.85	1.15E−03	2.99	7.89E−03	2.92
40 × 40	4.66E−03	3.00	9.85E−02	2.92	1.43E−04	3.01	1.01E−03	2.97
80 × 80	5.82E−04	3.00	1.26E−02	2.97	1.79E−05	3.00	1.30E−04	2.96

We set $\lambda = -13$ and choose an initial condition $u(x, y, 0) = 1$ and the boundary conditions $u(x, 0, t) = 6(8\mathrm{e}^{2x} + \mathrm{e}^{-2x})$, $u(x, 1, t) = (5\mathrm{e}^3 + \mathrm{e}^{-3})(8\mathrm{e}^{2x} + \mathrm{e}^{-2x})$, $u(0, y, t) = 9(5\mathrm{e}^{3y} + \mathrm{e}^{-3y})$ and $u(1, y, t) = (8\mathrm{e}^2 + \mathrm{e}^{-2})(5\mathrm{e}^{3y} + \mathrm{e}^{-3y})$. The exact steady state solution is $u(x, y) = (8\mathrm{e}^{2x} + \mathrm{e}^{-2x})(5\mathrm{e}^{3y} + \mathrm{e}^{-3y})$. The numerical results are given in Table 5.14. This time, since the exact solution is not in the finite element space $E^k(\alpha, \beta)$ no matter how we choose the parameters, we cannot expect errors at the round-off level. However, we can see that our method automatically choose suitable parameters in the exponential space $E^k(\alpha, \beta)$, since the magnitude of the errors are much smaller than those from the regular DG method with the same meshes.

6. Concluding remarks

We have designed and tested different approximation spaces for the discontinuous Galerkin (DG) and the local discontinuous Galerkin (LDG) methods. Our work demonstrates the flexibility of the DG and LDG methods with finite element spaces. We have formulated conditions under which the approximation results similar to those of polynomial spaces can be proven, and verified that several approximation spaces including the exponential spaces and trigonometric spaces satisfy these conditions. Stability and error estimates for the DG methods based on non-polynomial spaces, similar to those for the DG methods based on standard polynomial spaces, have been proven under the these conditions. We have also investigated methods to determine the parameters in the local approximation spaces dynamically, and have demonstrated through numerical examples the effectiveness of these methods to identify parameters suitable for the approximated solutions. Numerical examples indicate that, when the local approximation spaces are well chosen, the DG approximation can be much more accurate than that using the standard polynomial spaces.

The main objective of this paper is to demonstrate the flexible approach to the solution space for a discontinuous Galerkin method. For practical problems with complicated solutions, it is certainly a challenge to identify the suitable approximation spaces with an efficient numerical procedure. If an optimal approximation space is not correctly identified, one may gain little or even no improvement in accuracy over standard discontinuous Galerkin method using standard polynomials spaces. However, the order of accuracy will not be lost if the space satisfies certain sufficient conditions spelled out in this paper.

In the future we plan to study more systematically the issue of identifying optimal parameters in the local spaces dynamically, especially for multi-dimensional problems. We also plan to study other local approximation spaces, including those with multi-scale basis functions [1,13], in order to solve other PDEs including those in the multi-scale context.

Appendix A

In this appendix, we verify that the approximation spaces $E^k(\alpha)$ given by (3.1), the trigonometric spaces $T^k(\alpha)$ given by (3.3), and their two-dimensional versions (3.7) and (3.8), do satisfy the conditions (3.4) in one dimension or (3.9) in multi-dimensions. We also establish the existence of the "inverse" tensor $\bar{A}$ of a fourth order tensor A.

A.1. The $E^k(\alpha)$ space

We choose a local basis $v_i = \mathrm{e}^{\alpha_j(x-x_j)}(x-x_j)^i$ in this case. We then have

$$\begin{pmatrix} \mathrm{e}^{\alpha_j(x-x_j)} \\ \mathrm{e}^{\alpha_j(x-x_j)}(x-x_j) \\ \cdots \\ \mathrm{e}^{\alpha_j(x-x_j)}(x-x_j)^k \end{pmatrix} = \begin{pmatrix} 1 & \alpha_j & \cdots & \frac{\alpha_j^k}{k!} \\ 0 & 1 & \cdots & \frac{\alpha_j^{k-1}}{(k-1)!} \\ \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & \cdots & 1 \end{pmatrix} \begin{pmatrix} 1 \\ (x-x_j) \\ \cdots \\ (x-x_j)^k \end{pmatrix} + \begin{pmatrix} \frac{\alpha_j^{k+1}}{(k+1)!} \\ \frac{\alpha_j^k}{k!} \\ \cdots \\ \alpha_j \end{pmatrix} (x-x_j)^{k+1} + \mathrm{O}((\Delta x_j)^{k+2}).$$

Therefore, we have

$$A = \begin{pmatrix} 1 & \alpha_j & \cdots & \frac{\alpha_j^k}{k!} \\ 0 & 1 & \cdots & \frac{\alpha_j^{k-1}}{(k-1)!} \\ \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & \cdots & 1 \end{pmatrix}, \quad b = 2\begin{pmatrix} \frac{|\alpha_j|^{k+1}}{(k+1)!} \\ \frac{|\alpha_j|^k}{k!} \\ \cdots \\ |\alpha_j| \end{pmatrix}.$$

It is easy to obtain

$$A^{-1} = \begin{pmatrix} 1 & -\alpha_j & \cdots & \frac{(-\alpha_j)^k}{k!} \\ 0 & 1 & \cdots & \frac{(-\alpha_j)^{k-1}}{(k-1)!} \\ \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & \cdots & 1 \end{pmatrix}$$

and we can observe that both A^{-1} and b are independent of Δx_j. Hence condition (3.4) is satisfied.

A.2. The $T^1(\alpha)$ and $T^2(\alpha)$ spaces

We set the basis functions as $v_0 = 1$, $v_1 = \sin\alpha_j(x-x_j)$ and $v_2 = 1 - \cos\alpha_j(x-x_j)$. We have

$$\begin{pmatrix} 1 \\ \sin\alpha_j(x-x_j) \\ 1-\cos\alpha_j(x-x_j) \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \alpha_j & 0 \\ 0 & 0 & \alpha_j^2/2 \end{pmatrix} \begin{pmatrix} 1 \\ (x-x_j) \\ (x-x_j)^2 \end{pmatrix} + \begin{pmatrix} 0 \\ -\frac{\alpha_j^3}{6} \\ 0 \end{pmatrix} (x-x_j)^3 + \mathrm{O}((\Delta x_j)^4).$$

For the $T^1(\alpha)$ space, we have $A = \begin{pmatrix} 1 & 0 \\ 0 & \alpha_j \end{pmatrix}$ and $b = \begin{pmatrix} 0 \\ \frac{|\alpha_j|^3}{3} \end{pmatrix}$. From which we obtain $A^{-1} = \begin{pmatrix} 1 & 0 \\ 0 & 1/\alpha_j \end{pmatrix}$. For the $T^2(\alpha)$ space, we have $A = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \alpha_j & 0 \\ 0 & 0 & \alpha_j^2/2 \end{pmatrix}$ and $b = \begin{pmatrix} 0 \\ \frac{|\alpha_j|^3}{3} \\ |\alpha_j| \end{pmatrix}$. Then we obtain $A^{-1} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1/\alpha_j & 0 \\ 0 & 0 & 2/\alpha_j^2 \end{pmatrix}$. We can see that both A^{-1} and b are independent of Δx_j for both $T^1(\alpha)$ and $T^2(\alpha)$ spaces. Therefore condition (3.4) is satisfied. The verification for the general $T^k(\alpha)$ case with $k > 2$ is similar.

A.3. The "inverse" tensor $\bar{A}$ of a fourth order tensor A

We need to create a bijective mapping L from upper-triangular fourth order tensors to matrices. The mapping L is defined as

$$LA = B, \quad b_{ij} = a_{mnpq}$$

with $m, n \geqslant 0,\ m + n \leqslant k,\ p, q \geqslant 0,\ p + q \leqslant k,\ i = (m^2 + n^2 + 2mn + m + 3n)/2$ and $j = (p^2 + q^2 + 2pq + p + 3q)/2$.

This mapping can be easily verified to be both injective and surjective.

Then we can obtain B^{-1} if B is invertible and let

$$\bar{A} = L^{-1}B^{-1}.$$

For the tensor A in the proof of Proposition 3.3, the matrix B can be easily verified to be invertible.

We can also easily verify that for any matrix b and c, if $b : A = c$ holds, then $c : \bar{A} = b$ is valid.

A.4. The $E^k(\alpha, \beta)$ space

We take

$$v_{mn} = \mathrm{e}^{\alpha_i(x-x_i)+\beta_j(y-y_j)}(x-x_i)^m(y-y_j)^n, \quad m+n \leqslant k$$

in this case. We know that

$$v_{mn} = \sum_{m\leqslant p, n\leqslant q, p+q\leqslant k} \frac{\alpha_i^{p-m}\beta_j^{q-n}}{(p-m)!(q-n)!}(x-x_i)^p(y-y_j)^q + \sum_{p+q=k+1} \frac{\alpha_i^{p-m}\beta_j^{q-n}}{(p-m)!(q-n)!}(x-x_i)^p(y-y_j)^q + \mathrm{O}((\Delta x)^{k+1}).$$

Hence we have $A = (a_{mnpq})$ with

$$a_{mnpq} = \begin{cases} \frac{\alpha_i^{p-m}\beta_j^{q-n}}{(p-m)!(q-n)!}, & m \leqslant p, n \leqslant q, p+q \leqslant k, \\ 0, & \text{otherwise.} \end{cases}$$

and $b = (b_{mn})$ with $b_{mn} = \frac{(|\alpha_i+\beta_j|)^{k+1-m-n}}{(k+1-m-n)!}$. Then we obtain $\bar{\mathrm{A}} = (\bar{\mathrm{a}}_{mnpq})$ with

$$\bar{a}_{mnpq} = \begin{cases} \frac{(-\alpha_i)^{p-m}(-\beta_j)^{q-n}}{(p-m)!(q-n)!}, & m \leqslant p, n \leqslant q, p+q \leqslant k, \\ 0, & \text{otherwise.} \end{cases}$$

We can see that both $\bar{\mathrm{A}}$ and b are independent of Δx.

A.5. The $T^1(\alpha, \beta)$ and $T^2(\alpha, \beta)$ spaces

We set

$$\begin{cases} v_{00} = 1, \\ v_{10} = \sin\alpha_i(x - x_i), \\ v_{01} = \sin\beta_j(y - y_j), \\ v_{20} = 1 - \cos\alpha_i(x - x_i), \\ v_{11} = \sin\alpha_i(x - x_i)\sin\beta_j(y - y_j), \\ v_{02} = 1 - \cos\beta_j(y - y_j) \end{cases}$$

in this case. For the $T^1(\alpha, \beta)$ space, we have $A = (a_{mnpq})$ with

$$a_{mnpq} = \begin{cases} a_{0000} = 1, \\ a_{1010} = \alpha_i, \\ a_{0101} = \beta_j, \\ 0, & \text{otherwise.} \end{cases}$$

and $b = \begin{pmatrix} 0 & \frac{|\alpha_i|^3}{3} \\ \frac{|\beta_j|^3}{3} & 0 \end{pmatrix}$. Then we obtain $\bar{A} = (\bar{a}_{mnpq})$ with

$$a_{mnpq} = \begin{cases} a_{0000} = 1, \\ a_{1010} = 1/\alpha_i, \\ a_{0101} = 1/\beta_j, \\ 0, & \text{otherwise.} \end{cases}$$

For the $T^2(\alpha, \beta)$ space, we have $A = (a_{mnpq})$ with

$$a_{mnpq} = \begin{cases} a_{0000} = 1, \\ a_{1010} = \alpha_i, \\ a_{0101} = \beta_j, \\ a_{2020} = \alpha_i^2/2, \\ a_{1111} = \alpha_i\beta_j, \\ a_{0202} = \beta_j^2/2, \\ 0, & \text{otherwise.} \end{cases}$$

and $b = \begin{pmatrix} 0 & \frac{|\alpha_i|^3}{3} & |\alpha_i| \\ \frac{|\beta_j|^3}{3} & \frac{|\alpha_i\beta_j|^3}{9} & 0 \\ |\beta_j| & 0 & 0 \end{pmatrix}$. Then we obtain $\bar{A} = (\bar{a}_{mnpq})$ with

$$a_{mnpq} = \begin{cases} a_{0000} = 1, \\ a_{1010} = 1/\alpha_i, \\ a_{0101} = 1/\beta_j, \\ a_{2020} = 2/\alpha_i^2, \\ a_{1111} = 1/(\alpha_i\beta_j), \\ a_{0202} = 2/\beta_j^2, \\ 0, & \text{otherwise.} \end{cases}$$

We can see both $\bar{A}$ and b are independent of Δx for both $T^1(\alpha, \beta)$ and $T^2(\alpha, \beta)$ spaces.

References

[1] I. Babuska, G. Caloz, E. Osborn, Special finite element methods for a class of second order elliptic problems with rough coefficients, SIAM Journal on Numerical Analysis 31 (1994) 945–981.

[2] S. Christofi, The study of building blocks for essentially non-oscillatory (ENO) schemes, Ph.D. thesis, Division of Applied Mathematics, Brown University, 1996.

[3] B. Cockburn, Discontinuous Galerkin methods for convection-dominated problems, in: T.J. Barth, H. Deconinck (Eds.), High-Order Methods for Computational Physics, Lecture Notes in Computational Science and Engineering, vol. 9, Springer, Berlin, 1999, pp. 69–224.

[4] B. Cockburn, S. Hou, C.-W. Shu, The Runge–Kutta local projection discontinuous Galerkin finite element method for conservation laws IV: the multidimensional case, Mathematics of Computation 54 (1990) 545–581.

[5] B. Cockburn, F. Li, C.-W. Shu, Locally divergence-free discontinuous Galerkin methods for the Maxwell equations, Journal of Computational Physics 194 (2004) 588–610.

[6] B. Cockburn, S.-Y. Lin, C.-W. Shu, TVB Runge–Kutta local projection discontinuous Galerkin finite element method for conservation laws III: one dimensional systems, Journal of Computational Physics 84 (1989) 90–113.

[7] B. Cockburn, C.-W. Shu, TVB Runge–Kutta local projection discontinuous Galerkin finite element method for conservation laws II: general framework, Mathematics of Computation 52 (1989) 411–435.

[8] B. Cockburn, C.-W. Shu, The Runge–Kutta local projection P^1-discontinuous-Galerkin finite element method for scalar conservation laws, Mathematical Modelling and Numerical Analysis (M^2AN) 25 (1991) 337–361.

[9] B. Cockburn, C.-W. Shu, The Runge–Kutta discontinuous Galerkin method for conservation laws V: multidimensional systems, Journal of Computational Physics 141 (1998) 199–224.

[10] B. Cockburn, C.-W. Shu, The local discontinuous Galerkin method for time-dependent convection–diffusion systems, SIAM Journal on Numerical Analysis 35 (1998) 2440–2463.

[11] B. Cockburn, C.-W. Shu, Runge–Kutta Discontinuous Galerkin methods for convection-dominated problems, Journal of Scientific Computing 16 (2001) 173–261.

[12] S. Gottlieb, C.-W. Shu, E. Tadmor, Strong stability preserving high order time discretization methods, SIAM Review 43 (2001) 89–112.

[13] T.Y. Hou, X.H. Wu, A multiscale finite element method for elliptic problems in composite materials and porous media, Journal of Computational Physics 134 (1997) 169–189.

[14] G. Jiang, C.-W. Shu, On cell entropy inequality for discontinuous Galerkin methods, Mathematics of Computation 62 (1994) 531–538.

[15] M.K. Kadalbajoo, K.C. Patidar, Exponentially fitted spline in compression for the numerical solution of singular perturbation problems, Computers and Mathematics with Applications 46 (2003) 751–767.

[16] F. Li, C.-W. Shu, Locally divergence-free discontinuous Galerkin methods for MHD equations, Journal of Scientific Computing 22–23 (2005) 413–442.

[17] F. Li, C.-W. Shu, Reinterpretation and simplified implementation of a discontinuous Galerkin method for Hamilton-Jacobi equations, Applied Mathematics Letters 18 (2005) 1204–1209.

[18] F. Li, C.-W. Shu, A local-structure-preserving local discontinuous Galerkin method for the Laplace equation, Methods and Applications of Analysis, submitted for publication.

[19] Y.N. Reddy, P.P. Chakravarthy, An exponentially fitted finite difference method for singular perturbation problems, Applied Mathematics and Computation 154 (2004) 83–101.

[20] C.-W. Shu, S. Osher, Efficient implementation of essentially non-oscillatory shock-capturing schemes, Journal of Computational Physics 77 (1988) 439–471.