Mathematics

https://www.aimspress.com/journal/Math

AIMS Mathematics, 10(9): 22579–22597.
DOI: 10.3934/math.20251005
Received: 09 July 2025
Revised: 14 September 2025
Accepted: 24 September 2025
Published: 29 September 2025

Research article

Efficiency analysis of continuous and discontinuous Galerkin finite element methods

Hanifa Hanif[1,2,*], **Jan Nordström**[3,4,*] **and Arnaud. G. Malan**[1]

[1] InCFD Research Group, Department of Mechanical Engineering, University of Cape Town, Cape Town 7700, South Africa
[2] Department of Mathematics, Sardar Bahadur Khan Women's University, Quetta, Pakistan
[3] Department of Mathematics, Linköping University, SE–581 83 Linköping, Sweden
[4] Department of Mathematics, University of Johannesburg, Auckland Park 2006, Johannesburg, South Africa

* **Correspondence:** Email: hanifahanif@outlook.com; jan.nordstrom@liu.se.

Abstract: We present a comparative analysis of the continuous Galerkin (CG) and discontinuous Galerkin (DG) finite element (FE) methods, focusing on their efficiency in solving advection–diffusion problems in the Summation By Parts–Simultaneous Approximation Term (SBP–SAT) framework. Our findings provide insights into the practical advantages and limitations of each approach, guiding future applications in high–order numerical simulations of real world applications.

Keywords: stability, convergence; efficiency; summation by parts; continuous Galerkin method; discontinuous Galerkin method
Mathematics Subject Classification: 65M12, 65M22, 65M60

1. Introduction

Efficiency is a fundamental consideration in numerical methods when solving partial differential equations (PDEs). High–order numerical methods, such as finite difference (FD) [1, 2], finite volume (FV) [3, 4], flux reconstruction (FR) [5, 6], spectral element (SE) [7, 8], continuous Galerkin (CG) [9, 10], and discontinuous Galerkin (DG) [11, 12] methods, have been recognized for their potential to efficiently solve PDEs. In this work, we will compare the efficiency of CG and DG methods. The key difference between the CG and DG approaches is that CG requires the solution to remain continuous across element boundaries, whereas DG allows for discontinuities at those interfaces. The DG method has gained popularity due to its applicability to non-linear hyperbolic problems that involve sharp discontinuities [13]. On the other hand, the CG method has received relatively little attention, despite being both stable and highly efficient when boundary conditions are correctly implemented [14, 15].

Summation–by–Parts (SBP) operators (first proposed in [16]) are suitable for higher–order spatial discretizations of PDEs, as these accurately mimic integration by parts [17]. Specifically, the SBP framework facilitates the development of higher–order boundary treatments in both space and time, ensuring provable stability [18,19]. When using SBP operators, boundary conditions are often enforced weakly through penalty terms with the Simultaneous Approximation Term (SAT) method. The SBP–SAT approach, when combined with well posed boundary conditions for linear initial boundary value problems (IBVPs), ensures semi–discrete stability via the energy method [20, 21]. Up to now, most developments have been done in the FD setting and migrated over to the FE, FV, FR and SE methods.

In this work, we revisit the application of SBP in the FE framework and formulate provably stable SBP–SAT compliant schemes, specifically within the context of the CG and DG methods. In particular, we aim to compare these methods in terms of accuracy, stiffness, and efficiency. Our objective is to provide insights into the advantages and limitations of CG and DG, guiding their application in large–scale real world simulations.

The rest of the paper is structured as follows: Section 2 introduces the continuous analysis of the problem. This is followed by Section 3, which derives the single–element CG scheme. Multi–element stability is established for the CG method in Section 4. In Section 5, stability for various DG methods is discussed. The numerical performance of the schemes in terms of efficiency is presented in Section 6. Finally, in Section 7, concluding remarks are drawn.

2. The continuous problem

Consider the one–dimensional IBVP for advection–diffusion

$$\begin{aligned} u_t + au_x &= \epsilon u_{xx}, & 0 \leqslant x \leqslant 1, \quad & 0 \leqslant t \leqslant T,\\ au - \epsilon u_x &= g_0(t), & x = 0, \quad & 0 \leqslant t \leqslant T,\\ \epsilon u_x &= g_n(t), & x = 1, \quad & 0 \leqslant t \leqslant T,\\ u &= f(x), & 0 \leqslant x \leqslant 1, \quad & t = 0, \end{aligned} \tag{2.1}$$

where $a, \epsilon > 0$. The functions g_0, g_n, and f are the boundary and initial data of the problem. Multiplying (2.1) with the solution u and integrating over the domain leads us to

$$\frac{d}{dt}\|u^2\| + 2\epsilon\|u_x\|^2 = \frac{1}{a}\Big(g_0^2 - \epsilon^2(u_x)_0^2\Big) - \frac{1}{a}\Big((au_n - \epsilon(u_x)_n)^2 - g_n^2\Big) \leq \frac{1}{a}\Big(g_0^2 + g_n^2\Big). \tag{2.2}$$

Here we have used the L^2 inner product property $\|\phi^2\| = (\phi, \phi) = \int_0^1 \phi^2\,dx$. The energy rate (2.2) guarantees that the solution is bounded by the data, and it will be the target for our numerical stability proofs.

3. Single element CG method

We consider the one–dimensional domain $\Omega \in [-1, 1]$, as a single element. The mesh with $n + 1$ nodes is numbered from 0 to n. The trial solution is

$$u(x, t) = \mathcal{L}^\top(x)\mathbf{u}(t), \tag{3.1}$$

where $\mathbf{u}(t) = [u_0(t), u_1(t)\ldots, u_n(t)]^\top$, and $\mathcal{L}(x) = [\ell_0(x), \ell_1(x), \ldots, \ell_n(x)]^\top$ are the Lagrange basis polynomials, defined as

$$\ell_i(x) = \prod_{\substack{k=o \\ k\neq j}}^{n} \frac{x - x_k}{x_j - x_k}. \tag{3.2}$$

The approximations $u_x(x,t) \approx \mathcal{L}_x^\top(x)\mathbf{u}(t)$ and $u_{xx}(x,t) \approx \mathcal{L}_{xx}^\top(x)\mathbf{u}(t)$ follow naturally from (3.1). The first derivative of the vector of Lagrange polynomials at (x_i) is given as

$$\mathcal{L}_x^T(x_i) = [\ell_{x_0}(x_i),\ \ell_{x_1}(x_i),\ \ldots,\ \ell_{x_n}(x_i)], \tag{3.3}$$

where $\ell_{x_j} = d\ell_j/dx$ and similarly for the second derivative. Incorporating the approximation of u, u_x and u_{xx} into Eq. (2.1), and imposing weak boundary conditions leads to

$$\mathcal{L}^\top \mathbf{u}_t + a\mathcal{L}_x^\top \mathbf{u} = \epsilon \mathcal{L}_{xx}^\top \mathbf{u} + L. \tag{3.4}$$

Here L is the lifting operator, which represents the weakly imposed boundary conditions as follows

$$L = \sigma_0[(a\mathcal{L}^\top - \epsilon\mathcal{L}_x^\top)_{x=x_0}\mathbf{u} - g_0] + \sigma_n[\epsilon(\mathcal{L}_x^\top)_{x=x_n}\mathbf{u} - g_n], \tag{3.5}$$

where σ_0 and σ_1 are penalty coefficients that will be determined to ensure stability.

Next, we integrate (3.4) over the domain Ω with the Galerkin weighting, resulting in

$$\int_\Omega \mathcal{L}\mathcal{L}^\top\, dx\, \mathbf{u}_t + a\int_\Omega \mathcal{L}\mathcal{L}_x^\top\, dx\, \mathbf{u} = \epsilon \int_\Omega \mathcal{L}\mathcal{L}_{xx}^\top dx\, \mathbf{u} + \mathbf{SAT}, \tag{3.6}$$

with

$$\mathbf{SAT} = \int_\Omega \mathcal{L}L\, dx = \sigma_0[a\mathcal{L}_0\mathcal{L}_0^\top - \epsilon\mathcal{L}_0\mathcal{L}_{x_0}^\top\mathbf{u} - \mathcal{L}_0 g_0] + \sigma_n[\epsilon\mathcal{L}_n\mathcal{L}_{x_n}^\top\mathbf{u} - \mathcal{L}_n g_n]. \tag{3.7}$$

All integrations are carried out using Gauss–Lobatto quadrature, defined as

$$\int_\Omega f(x)\, dx = \int_{-1}^{1} f(\xi)\, d\xi = \sum_{i=0}^{n} w_i f(\xi_i), \tag{3.8}$$

where ξ_i and w_i represent the Gauss–Lobatto node coordinate and corresponding quadrature weights, respectively. The integral (3.8) is exact for polynomials $f(x)$ of degree $\leq (2n-1)$.

3.1. Discrete operators on SBP–SAT form

Let us start by formulating the mass matrix in Eq. (3.6) in the conventional SBP form

$$\mathcal{P} = \int_\Omega \mathcal{L}\mathcal{L}^\top\, dx = \sum_{i=0}^{n} w_i\mathcal{L}(x_i)\mathcal{L}^\top(x_i) = \text{diag}(w_0, w_1, \ldots, w_n). \tag{3.9}$$

Next, considering the advection term in Eq. (3.6) and integrating by parts gives us

$$Q_x = \int_\Omega \mathcal{L}\mathcal{L}_x^\top\, dx = \mathcal{L}\mathcal{L}^\top\Big|_{x=x_0}^{x=x_n} - \int_\Omega \mathcal{L}_x\mathcal{L}^\top\, dx, \tag{3.10}$$

where Q is the weak form of the first derivative. Equation (3.10) leads us to the so–called SBP property

$$\mathcal{B} = \mathcal{L}\mathcal{L}^\top\Big|_{x=x_0}^{x=x_n} = Q_x + Q_x^\top = \text{diag}(-1, 0, \ldots, 0, +1), \tag{3.11}$$

where Q is almost skew–symmetric. By identifying terms in (2.1) and (3.6), we define the strong form first derivative SBP operator as

$$D_x = \mathcal{P}^{-1}Q_x, \tag{3.12}$$

where the matrices $\mathcal{P}$ and Q_x are given in (3.9) and (3.10), respectively. Finally, we integrate the diffusion term over the domain such that

$$Q_{xx} = \int_\Omega \mathcal{L}\mathcal{L}_{xx}^\top \, dx = -\mathcal{A}_x + \mathcal{B}_x, \tag{3.13}$$

where

$$\mathcal{A}_x = \int_\Omega \mathcal{L}_x\mathcal{L}_x^\top \, dx, \quad \text{and} \quad \mathcal{B}_x = \mathcal{L}\mathcal{L}_x^\top\Big|_{x=x_0}^{x=x_n} = \begin{bmatrix} -\ell_{x_0}(x_0) & -\ell_{x_1}(x_0) & \cdots & -\ell_{x_n}(x_0) \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & 0 \\ \ell_{x_0}(x_n) & \ell_{x_1}(x_n) & \cdots & \ell_{x_n}(x_n) \end{bmatrix}. \tag{3.14}$$

Lemma 1. *The matrix $\mathcal{A}_x$ in* (3.14) *is positive semi–definite.*

Proof. Applying Gauss–Lobatto quadrature (3.8) to matrix $\mathcal{A}_x$ in (3.14) gives us (see Appendix A)

$$\mathcal{A}_x = D_x^\top \mathcal{P} D_x. \tag{3.15}$$

For an arbitrary vector $\mathbf{U}$, (3.15) gives us $\mathbf{U}^\top\mathcal{A}_x\mathbf{U} = (D_x\mathbf{U})^\top\mathcal{P}(D_x\mathbf{U}) = \mathbf{U}_x^\top\mathcal{P}\mathbf{U}_x = \|\mathbf{U}_x\|_\mathcal{P}^2 \geq 0$, where we have used $\mathbf{U}_x = D_x\mathbf{U}$. □

3.2. Discrete energy estimate

In the SBP–SAT framework, Eq (3.6) can be written as

$$\begin{aligned} \mathcal{P}\mathbf{u}_t + aQ_x\mathbf{u} &= \epsilon Q_{xx}\mathbf{u} + \sigma_0 E_0(a\mathbf{u} - \epsilon D_x\mathbf{u} - \mathbf{g}) + \sigma_1 E_n(\epsilon D_x\mathbf{u} - \mathbf{g}), \\ \mathbf{u}(0) &= \mathbf{f}_0, \end{aligned} \tag{3.16}$$

where $\mathbf{g} = [g_0(t), g_1(t), \ldots, g_n(t)]^\top, E_0 = \text{diag}(1, 0, \cdots, 0)$ and $E_n = \text{diag}(0, \cdots, 0, 1)$. Following [22], we introduce the proposition below.

Proposition 1. *If $\sigma_0 = \sigma_n = -1$, then the scheme* (3.16) *is energy stable.*

Proof. The energy method (multiplying (3.16) with $\mathbf{u}^T$ from the left) and Lemma 1 gives us

$$\mathbf{u}^\top\mathcal{P}\mathbf{u}_t + a\mathbf{u}^\top Q_x\mathbf{u} + \epsilon\mathbf{u}_x^\top\mathcal{P}\mathbf{u}_x = \epsilon\mathbf{u}^\top\mathcal{B}_x\mathbf{u} + \sigma_0\mathbf{u}^\top E_0(a\mathbf{u} - \epsilon D_x\mathbf{u} - \mathbf{g}_0) + \sigma_n\mathbf{u}^\top E_n(\epsilon D_x\mathbf{u} - \mathbf{g}_1), \tag{3.17}$$

where we have used the notation $\mathbf{u}_x = D_x\mathbf{u}$. Transposing (3.17), adding it to itself, and using the notation $||\mathbf{u}||^2_{\mathcal{P}} = \mathbf{u}^T\mathcal{P}\mathbf{u}$ and $||\mathbf{u}_x||^2_{\mathcal{P}} = \mathbf{u}_x^\top\mathcal{P}\mathbf{u}_x$ yields

$$\begin{aligned}\frac{d}{dt}||\mathbf{u}||^2_{\mathcal{P}} + 2\epsilon||\mathbf{u}_x||^2_{\mathcal{P}} = \frac{1}{a(1+2\sigma_0)}\Big(\big(a(1+2\sigma_0)u_0 + g_0\big)^2 - g_0^2\Big) - \frac{1}{a}\Big(\big(au_n + \sigma_n g_n\big)^2 \\ - \sigma_n^2 g_n^2\Big) - 2(1+\sigma_0)u_0 g_0 - 2(1+\sigma_0)\epsilon u_0 (u_x)_0 + 2(1+\sigma_n)\epsilon u_n (u_x)_n.\end{aligned} \tag{3.18}$$

With the choice $\sigma_0 = \sigma_n = -1$, Eq. (3.18) is reduced to

$$\frac{d}{dt}||\mathbf{u}||^2_{\mathcal{P}} + 2\epsilon||\mathbf{u}_x||^2_{\mathcal{P}} = \frac{1}{a}\Big(g_0^2 - \epsilon^2(u_x)_0^2\Big) - \frac{1}{a}\Big(\big(au_n - \epsilon(u_x)_n\big)^2 - g_n^2\Big) \le \frac{1}{a}\Big(g_0^2 + g_n^2\Big). \tag{3.19}$$

The bound (3.19) mimics (2.2) and guarantees that the approximation (3.16) is energy stable. □

4. Multi–element CG method

All operators have so far been developed on a single–element; next, we assemble the $(n+1)\times(n+1)$ matrices for the m elements to construct a global matrix of order $(mn+1)\times(mn+1)$. We illustrate the procedure by merging two elements. Let A^e be the element matrices such that the global matrix A combines the two single–element matrices in the following manner.

$$A = \sum_{\forall(e)} A^e = \begin{bmatrix} A^{(1)}_{00} & \cdots & A^{(1)}_{0n} & \cdots & 0 \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ A^{(1)}_{n0} & \cdots & A^{(1)}_{nn} + A^{(2)}_{00} & \cdots & A^{(2)}_{0n} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ 0 & \cdots & A^{(2)}_{n0} & \cdots & A^{(2)}_{nn} \end{bmatrix}. \tag{4.1}$$

Lemma 2. *If a single element operator A is positive semi–definite, then the global operator retains this positive semi–definite property.*

Proof. Consider two arbitrary vectors $\bar{U}_1 = [\bar{u}_0, \cdots, \bar{u}_n]^\top$ and $\bar{U}_2 = [\bar{u}_n, \cdots, \bar{u}_{2n}]^\top$ such that

$$\bar{W} = [\bar{u}_0, \cdots, \bar{u}_n, \cdots, \bar{u}_{2n}]^\top. \tag{4.2}$$

The merging property (4.1) for two single element positive semi–definite matrices A_1 and A_2 gives us

$$\begin{aligned}\bar{W}^\top A \bar{W} &= \begin{bmatrix} \bar{u}_0 \\ \vdots \\ \bar{u}_n \\ \vdots \\ \bar{u}_{2n} \end{bmatrix}^\top \begin{bmatrix} A^{(1)}_{00} & \cdots & A^{(1)}_{0n} & \cdots & 0 \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ A^{(1)}_{n0} & \cdots & A^{(1)}_{nn} + A^{(2)}_{00} & \cdots & A^{(2)}_{0n} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ 0 & \cdots & A^{(2)}_{n0} & \cdots & A^{(2)}_{nn} \end{bmatrix} \begin{bmatrix} \bar{u}_0 \\ \vdots \\ \bar{u}_n \\ \vdots \\ \bar{u}_{2n} \end{bmatrix}, \\ &= \begin{bmatrix} \bar{u}_0 \\ \vdots \\ \bar{u}_n \end{bmatrix}^\top \begin{bmatrix} A^{(1)}_{00} & \cdots & A^{(1)}_{0n} \\ \vdots & \ddots & \vdots \\ A^{(1)}_{n0} & \cdots & A^{(1)}_{nn} \end{bmatrix} \begin{bmatrix} \bar{u}_0 \\ \vdots \\ \bar{u}_n \end{bmatrix} + \begin{bmatrix} \bar{u}_n \\ \vdots \\ \bar{u}_{2n} \end{bmatrix}^\top \begin{bmatrix} A^{(2)}_{00} & \cdots & A^{(2)}_{0n} \\ \vdots & \ddots & \vdots \\ A^{(2)}_{n0} & \cdots & A^{(2)}_{nn} \end{bmatrix} \begin{bmatrix} \bar{u}_n \\ \vdots \\ \bar{u}_{2n} \end{bmatrix}, \\ &= \bar{U}_1^\top A_1 \bar{U}_1 + \bar{U}_2^\top A_2 \bar{U}_2 \ge 0,\end{aligned} \tag{4.3}$$

which concludes the proof. $\square$

Using (4.1), the global mass matrix remains diagonal and retains its positive–definite property since

$$\mathcal{P} = \text{diag}(w_0, w_1, \cdots, \underbrace{w_n + w_0}_{\text{shared node}}, w_1, \cdots, w_n). \tag{4.4}$$

Furthermore, using the merging process (4.1), all internal diagonals of the global matrix Q become zero. The global weak first derivative exhibits a global almost–skew–symmetric property: $Q_x + Q_x^\top = \text{diag}(-1, 0, \cdots, 0, +1) = \mathcal{B}$, which confirms that the global weak first derivative matrix is in SBP form. An example of a two element mesh for two quadratic elements is illustrated below: Let Q^L and Q^R be the local weak first derivative for the left and right quadratic elements as shown below

$$Q^L = \frac{1}{6}\begin{bmatrix} -3 & 4 & -1 \\ -4 & 0 & 4 \\ 1 & -4 & 3 \end{bmatrix} \quad \text{and} \quad Q^R = \frac{1}{6}\begin{bmatrix} -3 & 4 & -1 \\ -4 & 0 & 4 \\ 1 & -4 & 3 \end{bmatrix}. \tag{4.5}$$

Using the merging procedure (4.1) for Q^L and Q^R gives us

$$Q = \sum_{\forall(e)} Q^e = \frac{1}{6}\begin{bmatrix} -3 & 4 & -1 & 0 & 0 \\ -4 & 0 & 4 & 0 & 0 \\ 1 & -4 & 3-3 & 4 & -1 \\ 0 & 0 & -4 & 0 & 4 \\ 0 & 0 & 1 & -4 & 3 \end{bmatrix}. \tag{4.6}$$

The globally merged Q in (4.6) remains almost skew symmetric. Therefore, when merging the single element formulation into the multi–element formulation, the symmetry properties are automatically retained.

In the weak second derivative, the solution is influenced by the global boundary term

$$\mathcal{B}_x\mathbf{u} = [-D_x|_{x=x_0}\mathbf{u}, 0, \cdots, 0, D_x^L|_{x=x_n}\mathbf{u} - D_x^R|_{x=x_0}\mathbf{u}, 0, \cdots, 0, D_x|_{x=x_n}\mathbf{u}]. \tag{4.7}$$

Following [23], we impose $u_x^L = u_x^R$, where L and R denote the left and right elements adjacent to a shared node by eliminating the difference in the first derivatives at element interfaces. This transforms (4.7) to

$$\mathcal{B}_x = [-D_x|_{x=x_0}, 0, \cdots, 0, D_x|_{x=x_n}] = \mathcal{B}D_x. \tag{4.8}$$

The global weak second derivative is assembled as

$$Q_{xx} = \mathcal{B}D_x - \sum_{\forall(e)} \mathcal{A}_x. \tag{4.9}$$

Lemmas 1 and 2 establish that the merged operator $\mathcal{A}_x$ in (4.9) is positive semi–definite. This implies that the energy rate in the multi–element setting is

$$\frac{d}{dt}\sum_{\forall(e)} \|\mathbf{u}\|_{\mathcal{P}}^2 + 2\epsilon \sum_{\forall(e)} \|\mathbf{u}_x\|_{\mathcal{P}}^2 = \frac{1}{a}\left(g_0^2 - \epsilon^2(u_x)_0^2\right) - \frac{1}{a}\left((au_n - \epsilon(u_x)_n)^2 - g_n^2\right) \leq \frac{1}{a}\left(g_0^2 + g_n^2\right). \tag{4.10}$$

This completes the proof of the following proposition.

Proposition 2. *Stability of the single–element CG method* (3.17) *is preserved in the multi–element setting.*

5. The DG method

The discretization for DG is the same as for CG at the elemental level. However, in the global operators, the nodes at the interfaces of the elements are not merged. As a result, two distinct nodes occupy the same spatial position at an interface. The global matrix in DG is thus of order $(m(n+1)) \times (m(n+1))$ and is structured as follows

$$A = \sum_{\forall(e)} A^e = \begin{bmatrix} A_{00}^{(1)} & \cdots & A_{0n}^{(1)} & 0 & \cdots & 0 \\ \vdots & \ddots & \vdots & 0 & \ddots & 0 \\ A_{n0}^{(1)} & \cdots & A_{nn}^{(1)} & 0 & \cdots & 0 \\ 0 & \cdots & 0 & A_{00}^{(2)} & \cdots & A_{0n}^{(2)} \\ \vdots & \ddots & 0 & \vdots & \ddots & \vdots \\ 0 & \cdots & 0 & A_{n0}^{(2)} & \cdots & A_{nn}^{(2)} \end{bmatrix}. \tag{5.1}$$

Following (5.1), the global mass matrix is defined as

$$\mathcal{P} = \mathrm{diag}(w_0, w_1, \cdots, w_n, w_0, w_1, \cdots, w_n). \tag{5.2}$$

Due to the disjoint block structure of the global matrices, Q_x does not exhibit an almost–skew–symmetric property: $Q_x + Q_x^T = \mathrm{diag}(-1, 0, \cdots, 0, +1, -1, 0, \cdots, 0, +1) \neq \mathcal{B}$. To address this issue, it is necessary to introduce either an interface penalty term or a numerical flux at the interface [22, 24].

5.1. The Local DG (LDG) method

The LDG method [24] is the most widely adopted DG interface treatment strategy for advection–diffusion equations. It employs an auxiliary variable that transform the original equation into a system of first–order equations. To illustrate the procedure, we introduce a new variable $p = u_x$, and reformulate (2.1) as

$$u_t + au_x - \epsilon p_x = 0, \quad -\epsilon u_x + \epsilon p = 0. \tag{5.3}$$

Let $(\mathbf{u}, \mathbf{p})$ and $(\mathbf{v}, \mathbf{q})$ be the solution vectors associated with the left and right elements, respectively. The semi–discrete SBP–SAT approximation of (2.1) can then be defined as [22]:

$$\begin{aligned} \mathbf{u}_t + aD_x\mathbf{u} - \varepsilon D_x\mathbf{p} &= \sigma_0\mathcal{P}^{-1}e_0(au_0 - \epsilon(D_x\mathbf{u})_0 - g_0) + \sigma_1^L\mathcal{P}^{-1}e_n(u_n - v_0) + \sigma_2^L\mathcal{P}^{-1}e_n(p_n - q_0), && (5.4a)\\ -\varepsilon D_x\mathbf{u} + \varepsilon\mathbf{p} &= \sigma_3^L\mathcal{P}^{-1}e_n(u_n - v_0), \\ \mathbf{v}_t + aD_x\mathbf{v} - \varepsilon D_x\mathbf{q} &= \sigma_n\mathcal{P}^{-1}e_n(\epsilon(D_x\mathbf{v})_n - g_n) + \sigma_1^R\mathcal{P}^{-1}e_0(v_0 - u_n) + \sigma_2^R\mathcal{P}^{-1}e_0(q_0 - p_n), && (5.4b)\\ -\varepsilon D_x\mathbf{v} + \varepsilon\mathbf{q} &= \sigma_3^R\mathcal{P}^{-1}e_0(v_0 - u_n), \end{aligned}$$

where $e_0 = \left[1, 0, \cdots, 0\right]^\top$ and $e_n = \left[0, \cdots, 0, 1\right]^\top$ are vectors of size $(n+1)$. The unknown penalty parameters $\sigma_1^L, \sigma_1^R, \sigma_2^L, \sigma_2^R, \sigma_3^L$, and σ_3^R will be determined by stability requirements. Following [22], we introduce and prove the following proposition.

Proposition 3. *The interface treatment in the scheme* (5.4) *is stable for*

$$\sigma_1^L \leq \frac{a}{2}, \;\; \sigma_1^R = \sigma_1^L - a, \;\; \sigma_2^R = \epsilon + \sigma_2^L, \;\; \sigma_3^L = -\epsilon - \sigma_2^L, \;\; \textit{and} \;\; \sigma_3^R = -\sigma_2^L. \tag{5.5}$$

Proof. We employ the energy method by multiplying (5.4a) and (5.4b) by $\mathbf{u}^\top\mathcal{P}$ and $\mathbf{v}^\top\mathcal{P}$, respectively. Adding their transposes and summing the results leads to the following expression with the choice $\sigma_0 = \sigma_n = -1$

$$\frac{d}{dt}\Big(\|\mathbf{u}\|^2 + \|\mathbf{v}\|^2\Big) + 2\epsilon\Big(\|\mathbf{u}_x\|^2 + \|\mathbf{v}_x\|^2\Big) = \frac{1}{a}\Big(g_0^2 - \epsilon^2(u_x)_0^2\Big) \; - \frac{1}{a}\Big((au_n - \epsilon(u_x)_n)^2 - g_n^2\Big) + V^\top HV, \tag{5.6}$$

where $V = [u_n, p_n, v_0, q_0]^\top$, $p_n = (D_x\mathbf{u})_n$, $q_0 = (D_x\mathbf{v})_0$ and

$$H = \begin{bmatrix} -a + 2\sigma_1^L & \epsilon + \sigma_2^L + \sigma_3^L & -\sigma_1^L - \sigma_1^R & -\sigma_2^L - \sigma_3^R \\ \epsilon + \sigma_2^L + \sigma_3^L & 0 & -\sigma_2^R - \sigma_3^L & 0 \\ -\sigma_1^L - \sigma_1^R & -\sigma_2^R - \sigma_3^L & a + 2\sigma_1^R & -\epsilon + \sigma_2^R + \sigma_3^R \\ -\sigma_2^L - \sigma_3^R & 0 & -\epsilon + \sigma_2^R + \sigma_3^R & 0 \end{bmatrix}. \tag{5.7}$$

The conditions in (5.5) modifies (5.7) to

$$H = \begin{bmatrix} -a + 2\sigma_1^L & 0 & a - 2\sigma_1^L & 0 \\ 0 & 0 & 0 & 0 \\ a - 2\sigma_1^L & 0 & -a + 2\sigma_1^L & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}. \tag{5.8}$$

Substituting (5.8) into (5.6), we obtain

$$\begin{aligned}\frac{d}{dt}\Big(\|\mathbf{u}\|^2 + \|\mathbf{v}\|^2\Big) + 2\epsilon\Big(\|\mathbf{u}_x\|^2 + \|\mathbf{v}_x\|^2\Big) = \frac{1}{a}\Big(g_0^2 - \epsilon^2(u_x)_0^2\Big) \; - \frac{1}{a}\Big((au_n - \epsilon(u_x)_n)^2 - g_n^2\Big) \\ + (a - 2\sigma_1^L)[u_n - v_0]^2.\end{aligned} \tag{5.9}$$

The condition on σ_1^L given in (5.5) guarantees that (5.6) is bounded by the data, that is,

$$\frac{d}{dt}\Big(\|\mathbf{u}\|^2 + \|\mathbf{v}\|^2\Big) + 2\epsilon\Big(\|\mathbf{u}_x\|^2 + \|\mathbf{v}_x\|^2\Big) \le \frac{1}{a}\Big(g_0^2 + g_n^2\Big). \tag{5.10}$$

The bound (5.10) mimics (2.2) and (3.19) and ensures energy stability of (5.4). □

5.2. The Baumann–Oden DG (BODG) method

One can, of course, also keep the second order formulation. The semi discrete approximation of (2.1) using the Baumann–Oden (BO) method [22, 25] can be written as

$$\begin{aligned}\mathbf{u}_t + aD_x\mathbf{u} - \varepsilon D_{xx}\mathbf{u} &= \sigma_0\mathcal{P}^{-1}e_0(au_0 - \epsilon(D_x\mathbf{u})_0 - g_0) + \sigma_1^L\mathcal{P}^{-1}e_n(u_n - v_0) \\ &\quad + \sigma_2^L\mathcal{P}^{-1}e_n[(D_x\mathbf{u})_n - (D_x\mathbf{v})_0] + \sigma_3^L\mathcal{P}^{-1}D_x^\top e_n(u_n - v_0), \qquad (5.11a) \\ \mathbf{v}_t + aD_x\mathbf{v} - \varepsilon D_{xx}\mathbf{v} &= \sigma_n\mathcal{P}^{-1}e_n(\epsilon(D_x\mathbf{v})_n - g_n) + \sigma_1^R\mathcal{P}^{-1}e_0(v_0 - u_n) \\ &\quad + \sigma_2^R\mathcal{P}^{-1}e_0[(D_x\mathbf{v})_0 - (D_x\mathbf{u})_n] + \sigma_3^R\mathcal{P}^{-1}D_x^\top e_0(v_0 - u_n). \qquad (5.11b)\end{aligned}$$

Again, following [22], we introduce the following proposition.

Proposition 4. *The conditions* (5.5) *in Proposition 3 lead to stability of the BODG method* (5.11).

Proof. Applying the energy method introduced in Proposition 3 to (5.11) leads us to

$$\frac{d}{dt}\Big(\|\mathbf{u}\|^2 + \|\mathbf{v}\|^2\Big) + 2\epsilon\Big(\|\mathbf{u}_x\|^2 + \|\mathbf{v}_x\|^2\Big) = \frac{1}{a}\Big(g_0^2 - \epsilon^2 (u_x)_0^2\Big) - \frac{1}{a}\Big(\big(au_n - \epsilon(u_x)_n\big)^2 - g_n^2\Big) + W^\top H W, \quad (5.12)$$

where $W = [u_n, u_{x_n}, v_0, v_{x_0}]$ and H is given in (5.7). Consequently, the conditions (5.5) also ensure an energy estimate and stability for (5.11). □

Corollary 1. *The stability conditions* (5.5) *in Proposition 3, which ensure stability of the interface treatment in scheme* (5.4)*, also ensure stability of the BODG method* (5.11).

6. Numerical results

The CG and DG schemes will be assessed using the method of manufactured solution (MMS) by comparing accuracy, stiffness, and efficiency. In particular, we will investigate the stiffness due to eigenvalue positions. To facilitate numerical integration over a general physical domain $[x_0, x_n]$, we map the physical domain to $\xi \in [-1, 1]$ using the transformation given below

$$x(\xi) = \frac{(1-\xi)}{2}x_0 + \frac{(1+\xi)}{2}x_n, \quad \text{and} \quad J = \frac{dx}{d\xi} = \frac{x_n - x_0}{2}, \quad (6.1)$$

where J is the Jacobian of this transformation. This transformation ensures that integrals expressed in the reference space correspond to the physical space, preserving the numerical quadrature accuracy.

6.1. One–dimensional advection-diffusion problem

In this test case, we consider the unsteady advection–diffusion Eq (2.1). The accuracy of the scheme is validated with the MMS $u = \sin(\omega x)\cos(at)$. The initial and boundary data are modified to coincide with the MMS solution, and a relevant forcing function is added. Boundary conditions, with boundary data from the MMS, are enforced weakly through the SAT method. The discretization error is computed using the P–Norm:

$$\epsilon_{\mathcal{P}} = \sqrt{(\mathbf{u} - \mathbf{u}_{\text{Exact}})^\top \mathcal{P}(\mathbf{u} - \mathbf{u}_{\text{Exact}})}. \quad (6.2)$$

The order of convergence between two consecutive meshes is calculated using the formula:

$$O(\epsilon_{\mathcal{P}}) = \frac{\log \epsilon_{\mathcal{P}_2} - \log \epsilon_{\mathcal{P}_1}}{\log M_1 - \log M_2}, \quad (6.3)$$

where M denotes the number of nodes (grid points), and the subscripts 1 and 2 denote two distinct meshes. Time integration is performed using the fourth–order Runge–Kutta method, and the maximum eigenvalue ($|\lambda|_{max}$) is used to determine the time step (Δt), i.e., $\Delta t = 1/|\lambda|_{max}$. The outer SAT penalties $\sigma_0 = -1$ and $\sigma_n = -1$ are used consistently. For LDG and BODG methods, the interface penalty parameters are chosen to be as $\sigma_1^L = 0$ and $\sigma_2^L = -\epsilon/2$. The results are evaluated at $t = 1$.

The convergence rates for the CG, LDG, and BODG cases for the polynomial of degree 2 are presented in Table 1. The highest accuracy is observed with CG, followed by LDG and BODG. Furthermore, the number of nonzero entries n_{nz} in DG is higher than CG, which increases the computational cost. In Tables 2 to 4, the convergence rates are presented for polynomial degree $n = 3$ for different values of ϵ. As shown in Table 2, the CG method achieves the expected convergence rate rapidly for larger values of ϵ, whereas smaller diffusion coefficients require significantly finer mesh resolution to recover the expected order. However, DG methods achieve consistent fourth-order convergence regardless of the value of ϵ; see Tables 3 and 4. Overall, the CG method converges at order $n + 2$, the LDG method at order $n + 1$, and the BODG method at order n when n is even and at order $n + 1$ when n is odd [22, 24–26].

Table 1. Order of convergence for $u_t + au_x = \epsilon u_{xx}$ with $a = 1, \epsilon = 0.01$ and $n = 2$.

m	CG			LDG			BODG		
	n_{nz}	$\epsilon_{\mathcal{P}}$	$\mathcal{O}(\epsilon_{\mathcal{P}})$	n_{nz}	$\epsilon_{\mathcal{P}}$	$\mathcal{O}(\epsilon_{\mathcal{P}})$	n_{nz}	$\epsilon_{\mathcal{P}}$	$\mathcal{O}(\epsilon_{\mathcal{P}})$
240	1921	5.708e−08	–	4311	6.034e−07	–	4311	1.810e−05	–
300	2401	2.338e−08	4.01	5391	3.145e−07	2.92	5391	1.202e−05	1.84
360	2881	1.128e−08	4.01	6471	1.842e−07	2.93	6471	8.559e−06	1.86
420	3361	6.086e−09	4.01	7551	1.170e−07	2.94	7551	6.404e−06	1.88
480	3841	3.568e−09	4.00	8631	7.890e−08	2.95	8631	4.972e−06	1.90

Table 2. Order of convergence of CG for $u_t + au_x = \epsilon u_{xx}$ with $a = 1$ and $n = 3$.

m	n_{nz}	$\epsilon = 0.01$		$\epsilon = 0.005$		$\epsilon = 0.001$		$\epsilon = 0.0005$	
		$\epsilon_{\mathcal{P}}$	$\mathcal{O}(\epsilon_{\mathcal{P}})$	$\epsilon_{\mathcal{P}}$	$\mathcal{O}(\epsilon_{\mathcal{P}})$	$\epsilon_{\mathcal{P}}$	$\mathcal{O}(\epsilon_{\mathcal{P}})$	$\epsilon_{\mathcal{P}}$	$\mathcal{O}(\epsilon_{\mathcal{P}})$
240	3601	1.356e−10	–	2.692e−10	–	1.127e−09	–	1.582e−09	–
300	4501	4.446e−11	5.00	8.847e−11	4.99	3.923e−10	4.73	5.983e−10	4.36
360	5401	1.788e−11	5.00	3.561e−11	5.00	1.634e−10	4.81	2.648e−10	4.48
420	6301	8.275e−12	5.00	1.649e−11	5.00	7.732e−11	4.86	1.309e−10	4.57
480	7201	4.262e−12	4.97	8.471e−12	4.99	4.026e−11	4.89	7.040e−11	4.65

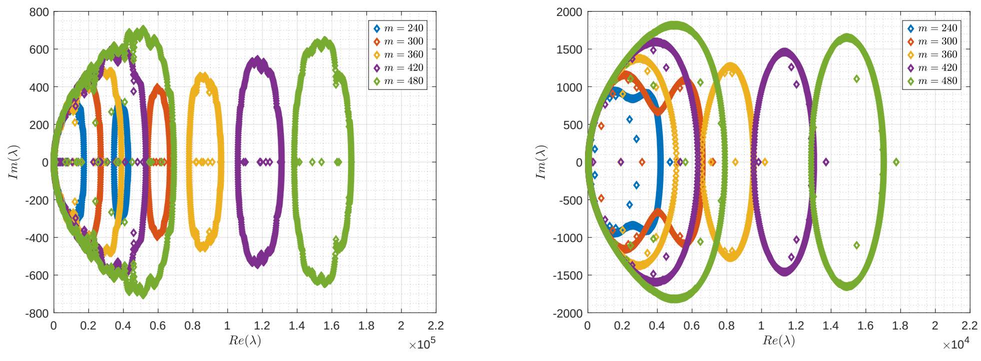
Table 3. Order of convergence of LDG for $u_t + au_x = \epsilon u_{xx}$ with $a = 1$ and $n = 3$.

m	n_{nz}	$\epsilon = 0.01$		$\epsilon = 0.005$		$\epsilon = 0.001$		$\epsilon = 0.0005$	
		$\epsilon_{\mathcal{P}}$	$\mathcal{O}(\epsilon_{\mathcal{P}})$	$\epsilon_{\mathcal{P}}$	$\mathcal{O}(\epsilon_{\mathcal{P}})$	$\epsilon_{\mathcal{P}}$	$\mathcal{O}(\epsilon_{\mathcal{P}})$	$\epsilon_{\mathcal{P}}$	$\mathcal{O}(\epsilon_{\mathcal{P}})$
240	6947	2.442e−09	–	2.505e−09	–	3.162e−09	–	3.451e−09	–
300	8687	9.946e−10	4.03	1.011e−09	4.07	1.243e−09	4.19	1.389e−09	4.08
360	10786	4.778e−10	4.02	4.825e−10	4.06	5.793e−10	4.19	6.551e−10	4.12
420	12167	2.572e−10	4.02	2.586e−10	4.05	3.042e−10	4.18	3.452e−10	4.16
480	13907	1.505e−10	4.02	1.508e−10	4.04	1.743e−10	4.17	1.977e−10	4.17

Table 4. Order of convergence of BODG for $u_t + au_x = \epsilon u_{xx}$ with $a = 1$ and $n = 3$.

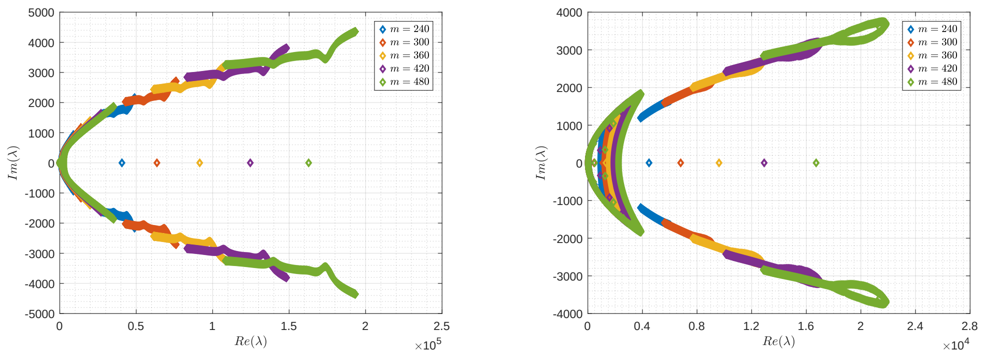
m	n_{nz}	$\epsilon = 0.01$		$\epsilon = 0.005$		$\epsilon = 0.001$		$\epsilon = 0.0005$	
		$\epsilon_{\mathcal{P}}$	$\mathcal{O}(\epsilon_{\mathcal{P}})$	$\epsilon_{\mathcal{P}}$	$\mathcal{O}(\epsilon_{\mathcal{P}})$	$\epsilon_{\mathcal{P}}$	$\mathcal{O}(\epsilon_{\mathcal{P}})$	$\epsilon_{\mathcal{P}}$	$\mathcal{O}(\epsilon_{\mathcal{P}})$
240	6947	1.495e−09	–	1.391e−09	–	1.138e−09	–	1.263e−09	–
300	8687	6.161e−10	3.97	5.765e−10	3.95	4.740e−10	3.93	4.817e−10	4.32
360	10427	2.983e−10	3.98	2.802e−10	3.96	2.330e−10	3.89	2.250e−10	4.17
420	12167	1.614e−10	3.98	1.522e−10	3.96	1.280e−10	3.89	1.203e−10	4.06
480	13907	9.483e−11	3.98	8.959e−11	3.97	7.611e−11	3.89	7.067e−11	3.98

In Figures 1 to 3, the global view of the spectrum is shown for the CG, LDG, and BODG cases, respectively, when $a = 1$ and $n = 3$. In all cases, the eigenvalues spread further along the real axis when the number of elements is increased. Additionally, as ϵ decreases, the spectrum contracts toward the origin along the positive real axis, indicating reduced stiffness in both CG and DG methods.



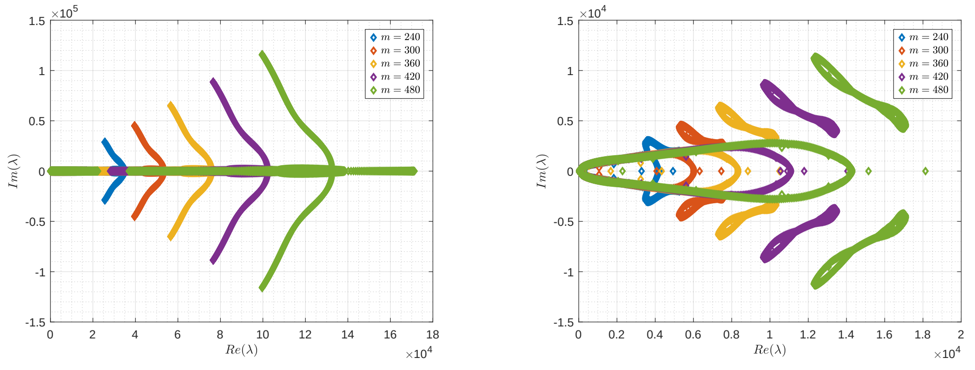
(a) $\epsilon = 0.01$ **(b)** $\epsilon = 0.001$

Figure 1. Global view of the eigenvalue spectrum for the CG method in one–dimension with $a = 1$ and $n = 3$.



(a) $\epsilon = 0.01$ **(b)** $\epsilon = 0.001$

Figure 2. Global view of the eigenvalue spectrum for the LDG method in one–dimension with $a = 1$ and $n = 3$.



(a) $\epsilon = 0.01$ **(b)** $\epsilon = 0.001$

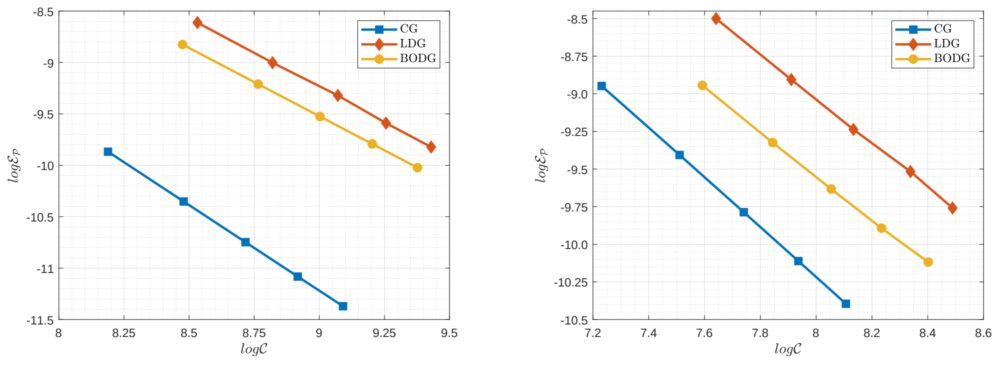
Figure 3. Global view of the eigenvalue spectrum for the BODG method in one–dimension with $a = 1$ and $n = 3$.

Table 5 shows that increasing the number of elements leads to spectral radius growth. The CG method consistently exhibits a smaller spectral radius compared to the DG methods. This shows that the CG formulation is inherently less stiff and more suitable for explicit time integration schemes.

Table 5. The spectral radius for $u_t + au_x = \epsilon u_{xx}$ with $a = 1$ and $n = 3$.

m	$\epsilon = 0.01$			$\epsilon = 0.001$		
	CG	LDG	BODG	CG	LDG	BODG
240	4.288e+04	4.902e+04	4.295e+04	4.730e+03	6.303e+03	5.628e+03
300	6.696e+04	7.616e+04	6.702e+04	7.200e+03	9.378e+03	8.040e+03
360	9.638e+04	1.093e+05	9.645e+04	1.019e+04	1.305e+04	1.086e+04
420	1.312e+05	1.483e+05	1.312e+05	1.371e+04	1.731e+04	1.410e+04
480	1.713e+05	1.934e+05	1.714e+05	1.775e+04	2.216e+04	1.813e+04

Figure 4 illustrates the relationship between computational cost and the error norm when $a = 1$ and $n = 3$. The computational cost C in each simulation is computed as $C = n_{nz} \times n_t$, where n_t represents the total number of iterations performed by the temporal solver. The results show that CG is more cost–effective than the DG variants.



(a) $\epsilon = 0.01$ **(b)** $\epsilon = 0.001$

Figure 4. Efficiency plot in the discrete L_2 for the one–dimensional case with $a = 1$ and $n = 3$.

6.2. *Two–dimensional advection–diffusion problem*

Consider the following two–dimensional steady advection–diffusion problem

$$\begin{aligned} au_x + bu_y &= \epsilon(u_{xx} + u_{yy}) + F, && 0 \leqslant x \leqslant 1, && 0 \leqslant y \leqslant 1,\\ au - \epsilon u_x &= g_0, && x = 0, && 0 \leqslant y \leqslant 1,\\ \epsilon u_x &= g_n, && x = 1, && 0 \leqslant y \leqslant 1,\\ bu - \epsilon u_y &= h_0, && y = 0, && 0 \leqslant x \leqslant 1,\\ \epsilon u_y &= h_n, && y = 1, && 0 \leqslant x \leqslant 1, \end{aligned} \tag{6.4}$$

where g_0, g_n, h_0 and h_n are the boundary data of the problem, and F is the source term. The accuracy of the scheme will be verified with the MMS $u = \sin(\omega(x + y))$, which produces the boundary data and source term. In this test case, we choose $a = b = 1$ and $\omega = 4\pi$. The outer SAT penalties $\sigma_0 = -1$

and $\sigma_n = -1$ are applied consistently in both spatial directions. For the LDG and BODG methods, the interface penalty parameters are set to $\sigma_1^L = 0$ and $\sigma_2^L = -\epsilon/2$ in both x– and y–directions. We need the total number of iterations to evaluate computational cost; therefore, we apply *pseudo–time stepping* to solve the steady–state problem [27, 28]. The solution is considered to have reached steady state when the discrete residual satisfies

$$\max\left(\frac{u^{k+1} - u^k}{\Delta t}\right) \leqslant 10^{-7}, \tag{6.5}$$

where Δt is the pseudo–time step, selected according to the stability condition, i.e., $\Delta t = 1/|\lambda|_{max}$. The convergence rates for the CG and DG methods are presented in Tables 6 and 7. As shown, both methods exhibit rates that align with the theoretical rates obtained in the one–dimensional case.

Table 6. Order of convergence for $au_x + bu_y = \epsilon(u_{xx} + u_{yy})$ with $\epsilon = 0.01$ and $n = 2$.

m	CG			LDG			BODG		
	n_{nz}	$\epsilon_{\mathcal{P}}$	$\mathcal{O}(\epsilon_{\mathcal{P}})$	n_{nz}	$\epsilon_{\mathcal{P}}$	$\mathcal{O}(\epsilon_{\mathcal{P}})$	n_{nz}	$\epsilon_{\mathcal{P}}$	$\mathcal{O}(\epsilon_{\mathcal{P}})$
16×16	7425	7.024e−03	–	24480	4.849e−03	–	24480	5.504e−03	–
20×20	11521	2.953e−03	3.99	38520	2.277e−03	3.39	38520	3.031e−03	2.67
24×24	16513	1.443e−03	4.02	59040	1.252e−03	3.28	55728	1.964e−03	2.38
28×28	22401	7.848e−04	4.03	76104	7.683e−04	3.17	76104	1.409e−03	2.15
32×32	29185	4.622e−04	4.03	99648	5.095e−04	3.08	99648	1.079e−03	2.00

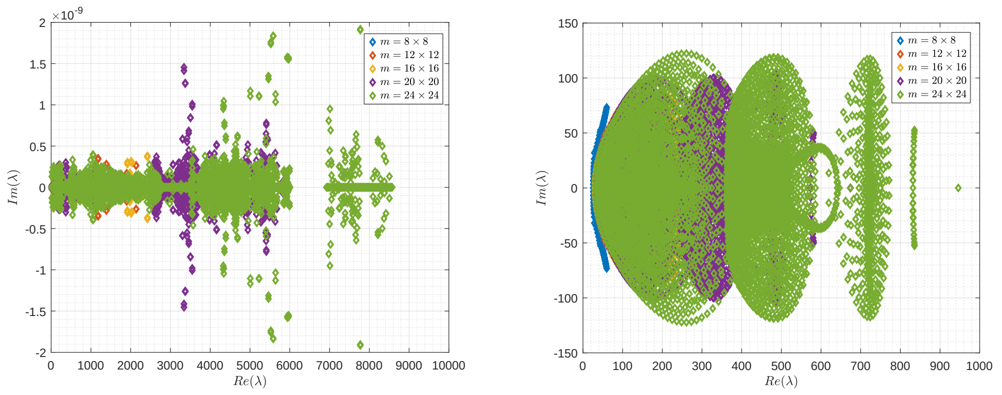
Table 7. Order of convergence for $au_x + bu_y = \epsilon(u_{xx} + u_{yy})$ with $\epsilon = 0.01$ and $n = 3$.

m	CG			LDG			BODG		
	n_{nz}	$\epsilon_{\mathcal{P}}$	$\mathcal{O}(\epsilon_{\mathcal{P}})$	n_{nz}	$\epsilon_{\mathcal{P}}$	$\mathcal{O}(\epsilon_{\mathcal{P}})$	n_{nz}	$\epsilon_{\mathcal{P}}$	$\mathcal{O}(\epsilon_{\mathcal{P}})$
8×8	5425	4.049e−03	–	12992	5.372e−03	–	12992	3.553e−03	–
12×12	12025	6.735e−04	4.58	29856	1.106e−03	3.90	29856	5.412e−04	4.64
16×16	21217	1.873e−04	4.56	53632	3.588e−04	3.91	53632	1.453e−04	4.57
20×20	33001	6.798e−05	4.63	84320	1.480e−04	3.97	84320	5.404e−05	4.43
24×24	47377	2.921e−05	4.70	121920	7.122e−05	4.01	121920	2.467e−05	4.30

The global view of the spectrum for the CG, LDG, and BODG in two–dimensions is shown in Figures 5 to 7, respectively, for $n = 3$. A similar behavior is observed as in the one–dimensional case. The spectral radius is presented numerically in Table 8. The CG method has a smaller spectral radius than the DG methods, which is again similar to the one–dimensional case.

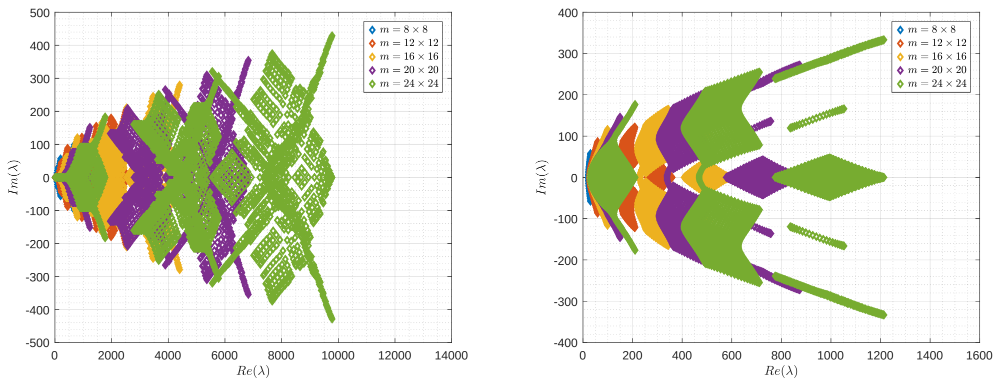
Table 8. The spectral radius for $au_x + bu_y = \epsilon(u_{xx} + u_{yy})$ with $n = 3$.

m	$\epsilon = 0.1$			$\epsilon = 0.01$		
	CG	LDG	BODG	CG	LDG	BODG
8×8	9.654e+02	1.145e+03	9.755e+02	1.314e+02	2.109e+02	2.022e+02
12×12	2.155e+03	2.514e+03	2.167e+03	2.670e+02	3.921e+02	3.782e+02
16×16	3.820e+03	4.410e+03	3.832e+03	4.479e+02	6.266e+02	5.930e+02
20×20	5.960e+03	6.835e+03	5.973e+03	6.741e+02	9.156e+02	8.429e+02
24×24	8.576e+03	9.790e+03	8.589e+03	9.460e+02	1.259e+03	1.125e+03



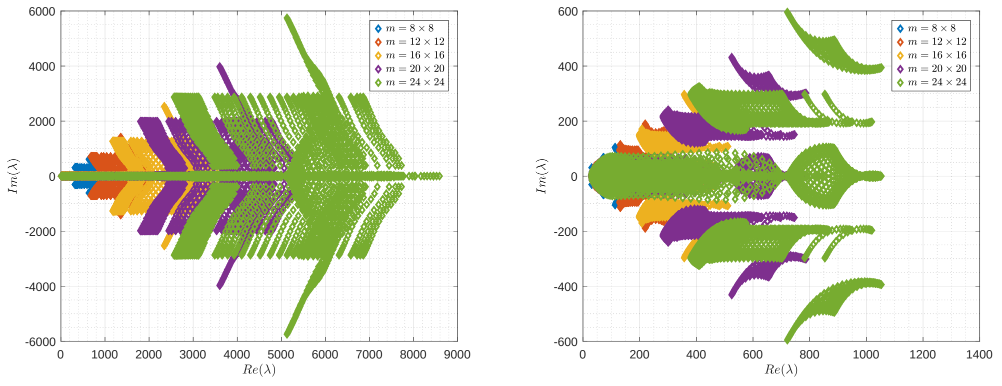
(a) $\epsilon = 0.1$ **(b)** $\epsilon = 0.01$

Figure 5. Global view of the eigenvalue spectrum for the 2D–CG method with $n = 3$.



(a) $\epsilon = 0.1$ **(b)** $\epsilon = 0.01$

Figure 6. Global view of the eigenvalue spectrum for the 2D–LDG method with $n = 3$.

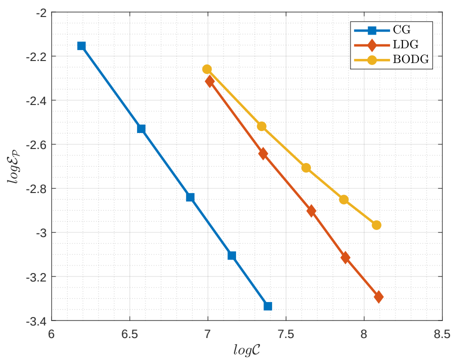


(a) $\epsilon = 0.1$ **(b)** $\epsilon = 0.01$

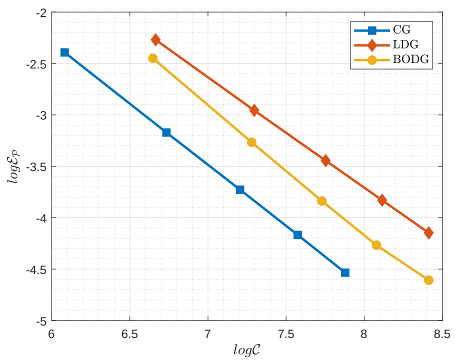
Figure 7. Global view of the eigenvalue spectrum for the 2D–BODG method with $n = 3$.

To compare the efficiency of the methods in two dimensions, the error norms are plotted against computational cost for $\epsilon = 0.01$; see Figure 8. The computational cost C in each simulation is computed in the same way as in one dimension. The results show that CG remains more cost–effective than the

DG methods. Overall, the pattern and behavior of the results are invariant with respect to the spatial dimension. In both one and two dimensions, CG remains more cost effective.



(a) $n = 2$



(b) $n = 3$

Figure 8. Efficiency plot in the discrete L_2 for the two–dimensional case with $\epsilon = 0.01$.

7. Conclusions

This paper examines the accuracy, stability and efficiency of the CG and DG schemes highlighting their differences in spectral behavior, accuracy and computational cost. Based on our findings, the CG method consistently exhibits a lower spectral radius, indicating reduced stiffness and making it more suitable for explicit time integration schemes. Additionally, CG achieves higher order accuracies while being less costly as compared to DG, making it the more efficient choice. These results highlight the advantages of CG in balancing accuracy and efficiency. They provide strong support for its suitability in large–scale simulations of advection–diffusion like IBVPs that require stable and cost–effective numerical formulations.

Author contributions

Hanifa Hanif: Investigation, Methodology, Software, Formal analysis, Validation, Writing–original draft; Jan Nordström: Conceptualization, Formal analysis, Supervision, Project administration, Writing–review and editing; Arnaud Malan: Conceptualization, Formal analysis, Supervision, Funding acquisition. All authors have read and approved the final version of the manuscript for publication.

Use of Generative-AI tools declaration

The authors declare that they have not used Artificial Intelligence (AI) tools in the creation of this article.

Acknowledgments

Jan Nordström was supported by Vetenskapsrådet, Sweden [no. 2021-05484 VR] and University of Johannesburg Global Excellence and Stature Initiative Funding. Arnaud Malan was financially wholly supported by the National Research Foundation (NRF) of South Africa grant number 89916.*

Conflict of interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

1. J. Nordström, J. Gong, E. Van der Weide, M. Svärd, A stable and conservative high order multi–block method for the compressible Navier–Stokes equations, *J. Comput. Phys.*, **228** (2009), 9020–9035. https://doi.org/10.1016/j.jcp.2009.09.005
2. M. Svärd, M. H. Carpenter, J. Nordström, A stable high–order finite difference scheme for the compressible Navier–Stokes equations, far–field boundary conditions, *J. Comput. Phys.*, **225** (2007), 1020–1038. https://doi.org/10.1016/j.jcp.2007.01.023
3. J. Nordström, K. Forsberg, C. Adamsson, P. Eliasson, Finite volume methods, unstructured meshes and strict stability for hyperbolic problems, *Appl. Numer. Math.*, **45** (2003), 453–473. https://doi.org/10.1016/S0168-9274(02)00239-8
4. J. Nordström, S. Eriksson, P. Eliasson, Weak and strong wall boundary procedures and convergence to steady–state of the Navier–Stokes equations, *J. Comput. Phys.*, **231** (2012), 4867–4884. https://doi.org/10.1016/j.jcp.2012.04.007
5. P. Castonguay, D. M. Williams, P. E. Vincent, A. Jameson, Energy stable flux reconstruction schemes for advection–diffusion problems, *Comput. Methods Appl. Mech. Eng.*, **267** (2013), 400–417. https://doi.org/10.1016/j.cma.2013.08.012
6. H. T. Huynh, A flux reconstruction approach to high–order schemes including discontinuous Galerkin methods, In: *18th AIAA Computational Fluid Dynamics Conference*, 2007, 4079. https://doi.org/10.2514/6.2007-4079
7. M. H. Carpenter, D. Gottlieb, Spectral methods on arbitrary grids, *J. Comput. Phys.*, **129** (1996), 74–86. https://doi.org/10.1006/jcph.1996.0234
8. M. H. Carpenter, T. C. Fisher, E. J. Nielsen, S. H. Frankel, Entropy stable spectral collocation schemes for the Navier–Stokes equations: Discontinuous interfaces, *SIAM J. Sci. Comput.*, **36** (2014), B835–B867. https://doi.org/10.1137/130932193
9. O. Karakashian, C. Makridakis, A space–time finite element method for the nonlinear Schrödinger equation: the continuous Galerkin method, *SIAM J. Numer. Anal.*, **36** (1999), 1779–1807. https://doi.org/10.1137/S0036142997330111
10. A. V. Idesman, A new high–order accurate continuous Galerkin method for linear elastodynamics problems, *Comput. Mech.*, **40** (2007), 261–279. https://doi.org/10.1007/s00466-006-0096-z
11. G. J. Gassner, A skew–symmetric discontinuous Galerkin spectral element discretization and its relation to SBP–SAT finite difference methods, *SIAM J. Sci. Comput.*, **35** (2013), A1233–A1253. https://doi.org/10.1137/120890144

*Any opinion, findings and conclusions or recommendations expressed in this material are those of the author(s) and therefore the NRF and DST do not accept any liability with regard thereto.

12. D. A. Kopriva, G. J. Gassner, J. Nordström, Stability of discontinuous Galerkin spectral element schemes for wave propagation when the coefficient matrices have jumps, *J. Sci. Comput.*, **88** (2021), 3. https://doi.org/10.1007/s10915-021-01516-w

13. M. Zakerzadeh, G. May, On the convergence of a shock capturing discontinuous Galerkin method for nonlinear hyperbolic systems of conservation laws, *SIAM J. Numer. Anal.*, **54** (2016), 874–898. https://doi.org/10.1137/14096503X

14. R. Abgrall, J. Nordström, P. Öffner, S. Tokareva, Analysis of the SBP–SAT stabilization for finite element methods part I: Linear problems, *J. Sci. Comput.*, **85** (2020), 43. https://doi.org/10.1007/s10915-020-01349-z

15. R. Abgrall, J. Nordström, P. Öffner, S. Tokareva, Analysis of the SBP–SAT stabilization for finite element methods part II: Entropy stability, *Commun. Appl. Math. Comput.*, **5** (2023), 573–595. https://doi.org/10.1007/s42967-020-00086-2

16. H. O. Kreiss, G. Scherer, Finite element and finite difference methods for hyperbolic partial differential equations, In: *Mathematical aspects of finite elements in partial differential equations*, 1974, 195–212. https://doi.org/10.1016/B978-0-12-208350-1.50012-1

17. K. Mattsson, J. Nordström, High order finite difference methods for wave propagation in discontinuous media, *J. Comput. Phys.*, **220** (2006), 249–269. https://doi.org/10.1016/j.jcp.2006.05.007

18. M. H. Carpenter, J. Nordström, D. Gottlieb, A stable and conservative interface treatment of arbitrary spatial accuracy, *J. Comput. Phys.*, **148** (1999), 341–365. https://doi.org/10.1006/jcph.1998.6114

19. J. Nordström, T. Lundquist, Summation–by–parts in time, *J. Comput. Phys.*, **251** (2013), 487–499. https://doi.org/10.1016/j.jcp.2013.05.042

20. D. C. Del Rey Fernández, J. E. Hicken, D. W. Zingg, Review of summation–by–parts operators with simultaneous approximation terms for the numerical solution of partial differential equations, *Comput. Fluids*, **95** (2014), 171–196. http://doi.org/10.1016/j.compfluid.2014.02.016

21. M. Svärd, J. Nordström, Review of summation–by–parts schemes for initial–boundary-value problems, *J. Comput. Phys.*, **268** (2014), 17–38. https://doi.org/10.1016/j.jcp.2014.02.031

22. J. Gong, J. Nordström, Interface procedures for finite difference approximations of the advection–diffusion equation, *J. Comput. Appl. Math.*, **236** (2011), 602–620. https://doi.org/10.1016/j.cam.2011.08.009

23. R. L. Taylor, O. C. Zienkiewicz, *The finite element method*, Oxford: Butterworth-Heinemann, 2013.

24. B. Cockburn, C. W. Shu, The local discontinuous Galerkin method for time–dependent convection–diffusion systems, *SIAM J. Numer. Anal.*, **35** (1998), 2440–2463. https://doi.org/10.1137/S0036142997316712

25. C. E. Baumann, J. T. Oden, A discontinuous hp finite element method for convection–diffusion problems, *Comput. Methods Appl. Mech. Eng.*, **175** (1999), 311–341. https://doi.org/10.1016/S0045-7825(98)00359-4

26. B. Cockburn, C. W. Shu, The Runge–Kutta discontinuous Galerkin method for conservation laws V: Multidimensional systems, *J. Comput. Phys.*, **141** (1998), 199–224. https://doi.org/10.1006/jcph.1998.5892

27. S. B. Hazara, V. Schulz, Simultaneous pseudo-timestepping for PDE-model based optimization problems, *Bit. Numer. Math.*, **44** (2004), 457–472. https://doi.org/10.1023/B:BITN.0000046815.96929.b8

28. J. Nordström, A. A. Ruggiu, Dual time-stepping using second derivatives, *J. Sci. Comput.*, **81** (2019), 1050–1071. https://doi.org/10.1007/s10915-019-01047-5

A. Appendix

Applying Gauss–Lobatto quadrature (3.8) to the weak first derivative Q_x defined in (3.10) gives us

$$
\begin{aligned}
Q_x &= \sum_{i=0}^{n} w_i \mathcal{L}(x_i)\mathcal{L}_x^\top(x_i),\\
&= w_0\mathcal{L}(x_0)\mathcal{L}_x^\top(x_0) + w_1\mathcal{L}(x_1)\mathcal{L}_x^\top(x_1) + \cdots + w_n\mathcal{L}(x_n)\mathcal{L}_x^\top(x_n),\\
&= w_0 \begin{bmatrix} \ell_0(x_0)\\ \ell_1(x_0)\\ \vdots\\ \ell_n(x_0)\end{bmatrix}\begin{bmatrix} \ell_{x_0}(x_0)\\ \ell_{x_1}(x_0)\\ \vdots\\ \ell_{x_n}(x_0)\end{bmatrix}^\top + w_1 \begin{bmatrix} \ell_1(x_1)\\ \ell_1(x_1)\\ \vdots\\ \ell_n(x_1)\end{bmatrix}\begin{bmatrix} \ell_{x_1}(x_1)\\ \ell_{x_1}(x_1)\\ \vdots\\ \ell_{x_n}(x_1)\end{bmatrix}^\top + \cdots + w_n \begin{bmatrix} \ell_n(x_n)\\ \ell_1(x_n)\\ \vdots\\ \ell_n(x_n)\end{bmatrix}\begin{bmatrix} \ell_{x_n}(x_n)\\ \ell_{x_1}(x_n)\\ \vdots\\ \ell_{x_n}(x_n)\end{bmatrix}^\top,\\
&= w_0 \begin{bmatrix} \ell_{x_0}(x_0) & \ell_{x_1}(x_0) & \cdots & \ell_{x_n}(x_0)\\ 0 & 0 & \cdots & 0\\ \vdots & \vdots & \ddots & \vdots\\ 0 & 0 & \cdots & 0\end{bmatrix} + w_1 \begin{bmatrix} 0 & 0 & \cdots & 0\\ \ell_{x_0}(x_1) & \ell_{x_1}(x_1) & \cdots & \ell_{x_n}(x_1)\\ \vdots & \vdots & \ddots & \vdots\\ 0 & 0 & \cdots & 0\end{bmatrix}\\
&+ \cdots + w_n \begin{bmatrix} 0 & 0 & \cdots & 0\\ 0 & 0 & \cdots & 0\\ \vdots & \vdots & \ddots & \vdots\\ \ell_{x_0}(x_n) & \ell_{x_1}(x_n) & \cdots & \ell_{x_n}(x_n)\end{bmatrix},\\
&= \begin{bmatrix} w_0 & 0 & \cdots & 0\\ 0 & w_1 & \cdots & 0\\ \vdots & \vdots & \ddots & \vdots\\ 0 & 0 & \cdots & w_n\end{bmatrix}\begin{bmatrix} \ell_{x_0}(x_0) & \ell_{x_1}(x_0) & \cdots & \ell_{x_n}(x_0)\\ \ell_{x_0}(x_1) & \ell_{x_1}(x_1) & \cdots & \ell_{x_n}(x_1)\\ \vdots & \vdots & \ddots & \vdots\\ \ell_{x_0}(x_n) & \ell_{x_1}(x_n) & \cdots & \ell_{x_n}(x_n)\end{bmatrix} = \mathcal{P}\mathbf{L}_x.
\end{aligned}
$$

The first derivative can thus be calculated as

$$D_x\mathbf{U} = \mathcal{P}^{-1}Q_x\mathbf{U} = \mathbf{L}_x\mathbf{U}. \tag{A1}$$

Next, applying Gauss–Lobatto quadrature (3.8) to matrix $\mathcal{A}_x$ in (3.14) gives us

$$\mathcal{A}_x = \sum_{i=0}^{n} w_i \mathcal{L}_x(x_i)\mathcal{L}_x^\top(x_i),$$

$$
\begin{aligned}
&= w_0\mathcal{L}(x_0)\mathcal{L}_x^\top(x_0) + w_1\mathcal{L}(x_1)\mathcal{L}_x^\top(x_1) + \cdots + w_n\mathcal{L}(x_n)\mathcal{L}_x^\top(x_n),\\
&= w_0 \begin{bmatrix} \ell_{x_0}(x_0) \\ \ell_{x_1}(x_0) \\ \vdots \\ \ell_{x_n}(x_0) \end{bmatrix} \begin{bmatrix} \ell_{x_0}(x_0) \\ \ell_{x_1}(x_0) \\ \vdots \\ \ell_{x_n}(x_0) \end{bmatrix}^\top + w_1 \begin{bmatrix} \ell_{x_1}(x_1) \\ \ell_{x_1}(x_1) \\ \vdots \\ \ell_{x_n}(x_1) \end{bmatrix} \begin{bmatrix} \ell_{x_1}(x_1) \\ \ell_{x_1}(x_1) \\ \vdots \\ \ell_{x_n}(x_1) \end{bmatrix}^\top + \cdots + w_n \begin{bmatrix} \ell_{x_n}(x_n) \\ \ell_{x_1}(x_n) \\ \vdots \\ \ell_{x_n}(x_n) \end{bmatrix} \begin{bmatrix} \ell_{x_n}(x_n) \\ \ell_{x_1}(x_n) \\ \vdots \\ \ell_{x_n}(x_n) \end{bmatrix}^\top,\\
&= w_0 \begin{bmatrix} \ell_{x_0}(x_0)\ell_{x_0}(x_0) & \ell_{x_0}(x_0)\ell_{x_1}(x_0) & \cdots & \ell_{x_0}(x_0)\ell_{x_n}(x_0) \\ \ell_{x_1}(x_0)\ell_{x_0}(x_0) & \ell_{x_1}(x_0)\ell_{x_1}(x_0) & \cdots & \ell_{x_1}(x_0)\ell_{x_n}(x_0) \\ \vdots & \vdots & \ddots & \vdots \\ \ell_{x_n}(x_0)\ell_{x_0}(x_0) & \ell_{x_n}(x_0)\ell_{x_1}(x_0) & \cdots & \ell_{x_n}(x_0)\ell_{x_n}(x_0) \end{bmatrix}\\
&+ w_1 \begin{bmatrix} \ell_{x_0}(x_1)\ell_{x_0}(x_1) & \ell_{x_0}(x_1)\ell_{x_1}(x_1) & \cdots & \ell_{x_0}(x_1)\ell_{x_n}(x_1) \\ \ell_{x_1}(x_1)\ell_{x_0}(x_1) & \ell_{x_1}(x_1)\ell_{x_1}(x_1) & \cdots & \ell_{x_1}(x_1)\ell_{x_n}(x_1) \\ \vdots & \vdots & \ddots & \vdots \\ \ell_{x_n}(x_1)\ell_{x_0}(x_1) & \ell_{x_n}(x_1)\ell_{x_1}(x_1) & \cdots & \ell_{x_n}(x_1)\ell_{x_n}(x_1) \end{bmatrix}\\
&+ \cdots + w_n \begin{bmatrix} \ell_{x_0}(x_n)\ell_{x_0}(x_n) & \ell_{x_0}(x_n)\ell_{x_1}(x_n) & \cdots & \ell_{x_0}(x_n)\ell_{x_n}(x_n) \\ \ell_{x_1}(x_n)\ell_{x_0}(x_n) & \ell_{x_1}(x_n)\ell_{x_1}(x_n) & \cdots & \ell_{x_1}(x_n)\ell_{x_n}(x_n) \\ \vdots & \vdots & \ddots & \vdots \\ \ell_{x_n}(x_n)\ell_{x_0}(x_n) & \ell_{x_n}(x_n)\ell_{x_1}(x_n) & \cdots & \ell_{x_n}(x_n)\ell_{x_n}(x_n) \end{bmatrix},\\
&= \begin{bmatrix} \ell_{x_0}(x_0) & \ell_{x_1}(x_0) & \cdots & \ell_{x_n}(x_0) \\ \ell_{x_0}(x_1) & \ell_{x_1}(x_1) & \cdots & \ell_{x_n}(x_1) \\ \vdots & \vdots & \ddots & \vdots \\ \ell_{x_0}(x_n) & \ell_{x_1}(x_n) & \cdots & \ell_{x_n}(x_n) \end{bmatrix}^\top \begin{bmatrix} w_0 & 0 & \cdots & 0 \\ 0 & w_1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & w_n \end{bmatrix} \begin{bmatrix} \ell_{x_0}(x_0) & \ell_{x_1}(x_0) & \cdots & \ell_{x_n}(x_0) \\ \ell_{x_0}(x_1) & \ell_{x_1}(x_1) & \cdots & \ell_{x_n}(x_1) \\ \vdots & \vdots & \ddots & \vdots \\ \ell_{x_0}(x_n) & \ell_{x_1}(x_n) & \cdots & \ell_{x_n}(x_n) \end{bmatrix},\\
&= D_x^\top \mathcal{P} D_x,
\end{aligned}
$$

where we have used (A1).