INTERNATIONAL JOURNAL OF
NUMERICAL ANALYSIS AND MODELING
Volume 14, Number 4-5, Pages 762–783

A NODAL SPARSE GRID SPECTRAL ELEMENT METHOD FOR MULTI-DIMENSIONAL ELLIPTIC PARTIAL DIFFERENTIAL EQUATIONS

ZHIJIAN RONG, JIE SHEN, AND HAIJUN YU*

Abstract. We develop a sparse grid spectral element method using nodal bases on Chebyshev-Gauss-Lobatto points for multi-dimensional elliptic equations. Since the quadratures based on sparse grid points do not have the accuracy of a usual Gauss quadrature, we construct the mass and stiffness matrices using a pseudo-spectral approach, which is exact for problems with constant coefficients and uniformly structured grids. Compared with the regular spectral element method, the proposed method has the flexibility of using a much less degree of freedom. In particular, we can use less points on edges to form a much smaller Schur-complement system with better conditioning. Preliminary error estimates and some numerical results are also presented.

Key words. Sparse grid, spectral element method, high-dimensional problem, adaptive method.

1. Introduction

Many scientific and engineering applications require solving high-dimensional partial differential equations (PDEs), e.g. the electronic Schrödinger equation [15], the Black-Scholes equation for option pricing [4], etc. Traditional methods for solving high-dimensional PDEs usually use tensor-product discretizations which need N^d total points if N points are used in each dimension. Such approximation quickly become unfeasible for problems with moderate to large dimensions, due to the so-called curse of dimensionality. However, for functions with special regularity, one can use sparse grid or hyperbolic cross approximation. This approach is first introduced by Smolyak for high-dimensional quadrature and interpolation problems in 1963 [20], and then extended for solving PDEs by Bungartz, Griebel et al. [6, 10, 7, 11, 2], where lower order finite element bases is used for problem with non-periodic boundary conditions and Fourier or wavelets bases are used for problem with periodic boundary conditions. For non-periodic spectral sparse grids, Barthelmann et al. [3] gave an error estimate for interpolation based on Chebyshev sparse grid, Shen and Wang [17] analyzed the approximation error of using hyperbolic cross Legendre approximation for solving standard elliptic PDEs. Shen and Yu developed efficient algorithms using sparse grid based on Chebyshev-Gauss-Lobatto (CGL) points for solving elliptic PDEs with non-periodic boundary conditions in bounded domains [18] and unbounded domains [19].

Even though sparse grids are developed for dealing with high dimensional problems, they are most effective for problems with some special properties. For example, Yserentant proved that the wave function of electronic Schrödinger equation has bounded mixed derivatives, which can be well approximated by sparse grids

Received by the editors on January 13, 2017, accepted on March 18, 2017.
2000 *Mathematics Subject Classification.* 65N35, 65N30.
*Corresponding author.

[21, 22]. However, many factors can contribute to the singular behavior of solutions. To deal with these and more general singularities, we construct in this paper a sparse grid Spectral Element Method (sgSEM) for high-dimensional PDEs. By decomposing the computational domain into sub-domains, we can effectively deal with many types of singularities, such as those in initial conditions, forcing terms and coefficients of the PDEs, and achieve a better convergence rate than usual Sparse Grid Methods (SGM) and Spectral Element Methods (SEM).

The rest of this paper is organized as follows. In Section 2, we briefly review some basic setup and results of the usual sparse grid methods. Then in Section 3, we construct the sgSEM for a model elliptic equation with variable coefficients and describe its basic properties. In Section 4, we develop a modified sgSEM (which we name it sgSEMm) with less nodes on edges, so as to reduce the size and condition number of the Schur-complement for the resulting linear algebraic system. Preliminary error estimates of sgSEM is given in Section 5 and numerical results for several examples are presented in Section 6 to verify the convergence and efficiency of the proposed methods. We end the paper with a few concluding remarks.

2. A brief review of sparse grid spectral methods

In order to present our sparse grid spectral element method, we need to first introduce the sparse grid spectral method. We start with some notations. We use bold letters to denote a d-dimensional vector. For d-dimensional coordinates, we use superscripts, e.g., $\boldsymbol{x} = (x^1, \ldots, x^d)$, to denote its components, and use subscripts, e.g., x_j, to denote interpolation points. For d-dimensional vectors other than coordinates, we use subscripts to denote its components. When $d = 1$, we have $x = x^1$.

2.1. Sparse grids and fast spectral transforms on sparse grids. Sparse grid method provides a feasible approximation for some high-dimensional functions. To introduce the sparse grid, we consider approximation of a high dimensional function $f(\boldsymbol{x}) : I^d \to \mathbb{R}$, where $I := [-1, 1]$. Suppose we have a one-dimensional interpolation scheme on a set of nested grids $\mathcal{X}_l : l = 0, 1, \ldots$, where $\mathcal{X}_l \subset \mathcal{X}_{l'}$, if $l < l'$. Denote $m_l = \mathrm{Card}\{\mathcal{X}_l\}$ and

$$\mathcal{X}_l = \{\, x_j : j \in \mathcal{I}_l \,\}, \quad \mathcal{I}_l := \{\, 0, \ldots, m_l - 1 \,\}. \tag{1}$$

The interpolation of level l is defined as

$$\mathcal{U}_l(f) := \sum_{k \in \mathcal{I}_l} \hat{f}_k^l \phi_k(x), \quad s.t. \quad \mathcal{U}_l(f)(x_j) = f(x_j), \quad \forall j \in \mathcal{I}_l, \tag{2}$$

where $\phi_k(x), k \in \mathcal{I}_l$ are the basis functions of interpolation space $V_l = \mathrm{span}\{\phi_k : k \in \mathcal{I}_l\}$ and $\hat{f}_k^l, k \in \mathcal{I}_l$ are the spectral coefficients. The above scheme is well defined if the matrix $\Psi = \big(\phi_k(x_j)\big)_{k,j \in \mathcal{I}_l}$ is non-singular.

If one extends the one-dimensional scheme to d-dimension by tensor product rule, then the total number of points will be $[m_l]^d$ which grows exponentially with d, leading to the so called curse of dimensionality. To reduce the number of points, Smolyak [20] introduced the so-called sparse grid approximation:

$$\mathcal{U}_l^d(f) := \sum_{|\boldsymbol{l}|_1 \leqslant l} \hat{\mathcal{U}}_{l_1} \otimes \hat{\mathcal{U}}_{l_2} \otimes \cdots \otimes \hat{\mathcal{U}}_{l_d}(f), \tag{3}$$

where $\boldsymbol{l} = (l_1, \ldots, l_d)$, $|\boldsymbol{l}|_1 = l_1 + l_2 + \cdots + l_d$, $\hat{\mathcal{U}}_l = \mathcal{U}_l - \mathcal{U}_{l-1}$, and $\mathcal{U}_{-1} = 0$. Define

$$\mathcal{X}_l^d := \bigcup_{|\boldsymbol{l}|_1 \leqslant l} \mathcal{X}_{l_1} \otimes \mathcal{X}_{l_2} \otimes \cdots \otimes \mathcal{X}_{l_d}, \tag{4}$$

$$\mathcal{I}_l^d := \bigcup_{|\boldsymbol{l}|_1 \leqslant l} \mathcal{I}_{l_1} \otimes \mathcal{I}_{l_2} \otimes \cdots \otimes \mathcal{I}_{l_d} \tag{5}$$

and $V_l^d = \{\, \phi_{\boldsymbol{j}} : \boldsymbol{j} = (j_1, \ldots, j_d) \in \mathcal{I}_l^d \,\}$. Then we have $\mathcal{X}_l^d = \{\boldsymbol{x}_{\boldsymbol{j}} = (x_{j_1}^1, \ldots, x_{j_d}^d) : \boldsymbol{j} = (j_1, \ldots, j_d) \in \mathcal{I}_l^d\}$, and the d-dimensional sparse grid interpolation $\mathcal{U}_l^d(f)$ is

$$\mathcal{U}_l^d(f)(\boldsymbol{x}) = \sum_{\boldsymbol{k} \in \mathcal{I}_l^d} \hat{f}_{\boldsymbol{k}}^l \phi_{\boldsymbol{k}}(\boldsymbol{x}), \tag{6}$$

such that

$$\mathcal{U}_l^d(f)(\boldsymbol{x}_{\boldsymbol{j}}) = f(\boldsymbol{x}_{\boldsymbol{j}}), \quad \forall \boldsymbol{j} \in \mathcal{I}_l^d, \tag{7}$$

where $\phi_{\boldsymbol{k}}(\boldsymbol{x}) = \phi_{k_1}(x^1) \cdots \phi_{k_d}(x^d)$ are d-dimensional basis functions by tensor-product.

Directly using equation (3) or solving equation (6) and (7) to construct sparse grid interpolation is not efficient. Hallatschek [13] built an efficient algorithm to calculate the Fast Fourier Transform on sparse grids (sgFFT), in which $I = S^1, \phi_k = e^{ikx}$. Shen and Yu [18] extended this algorithm to Fast Chebyshev Transform on sparse grids (sgFCT) and Fast General Transforms on sparse grids (sgFGT). The basic idea of those fast algorithms is to use one-dimensional hierarchical bases $\tilde{\phi}_k(x)$, defined by

$$\tilde{\phi}_k(x) \in V_{l(k)} \quad \text{s.t.} \quad \tilde{\phi}_k(x_j) = 0, \quad \forall\, j \in \mathcal{I}_{l(k)-1}, \tag{8}$$

where $l(k) = \min\{l : k \in \mathcal{I}_l\}$. Then the sparse grid interpolation can be reformulated as

$$\mathcal{U}_l^d(f) = \sum_{\boldsymbol{k} \in \mathcal{I}_l^d} \hat{f}_{\boldsymbol{k}} \tilde{\phi}_{\boldsymbol{k}}(\boldsymbol{x}), \tag{9}$$

where the coefficients $\{\hat{f}_{\boldsymbol{k}}, \boldsymbol{k} \in \mathcal{I}_l^d\}$ are determined by the interpolation property (7). The advantage of using hierarchical bases is that the hierarchical coefficients $\hat{f}_{\boldsymbol{k}}$ in equation (9) do not depend on level number l, which means that they can be calculated by using any tensor-product grid that contains enough information. Furthermore, the hierarchical bases $\tilde{\phi}_k(x),\ k \in \mathcal{I}_l \backslash \mathcal{I}_{l-1}$ are linear combinations of $\{\phi_j(x), j \in \mathcal{I}_l\}$. Moreover, for the Fourier and Chebyshev bases, the combinations are also sparse. So a transform between the representation using original spectral bases and the representation using hierarchical bases can be performed with linear computational cost by using the sgFFT and sgFCT algorithms in the following two steps [18]:

- The first step is the transform from function values at sparse grid points $\{\, f(\boldsymbol{x}_{\boldsymbol{j}}), \boldsymbol{j} \in \mathcal{I}_l^d \,\}$ to hierarchical spectral coefficients $\{\, \hat{f}_{\boldsymbol{k}}, \boldsymbol{k} \in \mathcal{I}_l^d \,\}$, which is accomplished by d consecutive transforms in one dimension on all the longest one-dimensional grids in the sparse grid structure using the underlying one-dimensional FFT or FCT followed by a one-dimensional transform from spectral coefficients to hierarchical coefficients.

- The second step is the d consecutive transforms from hierarchical coefficients $\{\hat{f}_{\boldsymbol{k}}, \boldsymbol{k} \in \mathcal{I}_l^d\}$ to spectral coefficients $\{\hat{f}_{\boldsymbol{k}}^l, \boldsymbol{k} \in \mathcal{I}_l^d\}$.

The inverse transforms can be done in a similar way. These fast transforms, including the matrix-vector product algorithm in solving PDEs, can be reformulated as a general tensor-product operations on sparse grid structure

$$c_{\boldsymbol{k}} = \sum_{\boldsymbol{j} \in \mathcal{I}_l^d} b_{\boldsymbol{j}} t_{j_1 k_1}^{(1)} t_{j_2 k_2}^{(2)} \cdots t_{j_d k_d}^{(d)}, \quad \boldsymbol{k} \in \mathcal{I}_l^d, \tag{10}$$

where $(t_{j_i,k_i}^{(i)})_{j_i,k_i}$, $i = 1, \ldots, d$ are the transform matrices for each dimension. It turns out that as long as those transform matrices are block tridiagonal matrices (the blocks should be consistent to the grid sizes $m_0, m_1, \ldots, m_l$), the above transforms, which we name as sparse grid fast general transform (sgFGT), can be performed similarly as sgFFT and sgFCT. When those transform matrices are not block tridiagonal, one can use sgFGT with a $L + U$ or $L \cdot U$ decomposition for the transform matrices. We refer to [18] and [19] for more detail.

2.2. Adaptivity on sparse grids. There are in general three types of adaptivity. One is h-adaptivity where one refines/coarsens mesh size, one is p-adaptivity where one increases/decreases the polynomial order in each element, and another is the so called r-adaptivity where one moves the meshes while keeping the number of unknowns fixed. Local h-adaptivity requires the use of basis functions with compact support, e.g. the finite element basis functions, wavelets etc. One advantage of using bases with local support is that we can refine the grids according to the local singularity of the solution. However, this will increase the complexity of programming, and the order of convergence will be finite. So here we do not consider local sparse grid adaptivity, instead we consider p-adaptivity within the framework of domain decomposition.

Denote $\hat{\mathcal{X}}_{\boldsymbol{l}} := (\mathcal{X}_{l_1} \backslash \mathcal{X}_{l_1-1}) \otimes \cdots \otimes (\mathcal{X}_{l_d} \backslash \mathcal{X}_{l_d-1})$, then these fast transforms (sgFFT, sgFCT and sgFGT) are available for all grids $\mathcal{X}_q^d$ with property

$$\text{if } \hat{\mathcal{X}}_{\boldsymbol{l}} \subseteq \mathcal{X}_q^d, \quad \text{then } \hat{\mathcal{X}}_{\boldsymbol{l}'} \subseteq \mathcal{X}_q^d, \ \forall\, \boldsymbol{l}' \leqslant \boldsymbol{l}, \tag{11}$$

where $\boldsymbol{l}' \leqslant \boldsymbol{l}$ stands for $l_i' \leqslant l_i$, for $i = 1, \ldots, d$. This means we do not have to use equation (4), (3) to construct the grid and corresponding approximation. Instead, we can adaptively refine the grid following the rule defined in equation (11) by examining the convergence of spectral coefficients $\{\hat{f}_{\boldsymbol{k}}, \boldsymbol{k} \in \mathcal{I}_l^d\}$.

Another way to introduce adaptivity is to choose appropriate values of $\{ m_0, m_1, \ldots, m_l \}$ based on approximation property of the bases.

2.3. A spectral sparse grid solver for elliptic PDEs. We describe below a sparse grid solver based on CGL points for PDEs on domain $I^d := [-1, 1]^d$. Note that the sparse grid solver based on Fourier basis for periodic elliptic equations is quite straightforward.

Consider a model problem

$$-\Delta u(\boldsymbol{x}) + \kappa u(\boldsymbol{x}) = f(\boldsymbol{x}), \quad \boldsymbol{x} \in I^d, \tag{12}$$

$$u(\boldsymbol{x})|_{\partial I^d} = 0, \tag{13}$$

where κ is a given constant. The corresponding weak form is

$$\text{Find } u \in H_0^1(I^d), \quad s.t. \quad (\nabla u, \nabla v) + \kappa(u, v) = (f, v), \quad \forall\, v \in H_0^1(I^d). \tag{14}$$

Let $X_n = V_l^d \cap H_0^1(I^d)$. Then a sparse grid Galerkin approximation of the weak form is [18]:

$$\text{Find } u_n \in X_n,\ s.t.\ (\nabla u_n, \nabla v_n) + \kappa(u_n, v_n) = (\mathcal{U}_l^d(f), v_n),\ \forall\, v_n \in X_n. \tag{15}$$

Note that the right hand side $(\mathcal{U}_l^d(f), v_n)$ of equation (15) is adopted for the reason of fast evaluation. Since the above formulation uses the usual inner product with uniform weight, it is natural to use Legendre polynomials as basis functions. However Legendre-Gauss (LG) quadrature points are not nested so we can not construct efficient Legendre-Gauss sparse grids. Fortunately, there are many works on using Legendre bases with Chebyshev-Gauss-Lobatto (CGL) quadrature points. For example, Chebyshev–Legendre solvers for PDEs are proposed in [9] and [16], some fast transforms between Chebyshev and Legendre bases are construct in [1] and [5]. Based on those works, efficient Chebyshev–Legendre sparse grid method for elliptic equations (12) can be constructed using quasi-orthogonal Legendre bases on sparse grid CGL points $\mathcal{X}_l^d$. Since the approximation space for function f, V_l^d, and the approximation space for the solution u, X_n, are different, two sets of bases are used:

- $\{T_{\boldsymbol{k}}(\boldsymbol{k}) := T_{k_1}(x_1)\cdots T_{k_d}(x_d) : \boldsymbol{k} \in \mathcal{I}_l^d\}$ are used for the interpolation space V_l^d;
- $\{\varphi_{\boldsymbol{k}}(\boldsymbol{x}) = \varphi_{k_1}(x_1)\cdots\varphi_{k_d}(x_d) : \boldsymbol{k} \in \mathcal{J}_l^d, \varphi_k(x) = L_k(x) - L_{k-2}(x)\}$ are used for solution space $X_n = V_l^d \cap H_0^1(I^d)$,

where $T_k(x)$, $L_k(x)$ are Chebyshev and Legendre polynomials of degree k, correspondingly, and

$$\mathcal{J}_l = \mathcal{I}_l \backslash \{0, 1\}, \quad \mathcal{J}_l^d := \bigcup_{|\boldsymbol{l}|_1 \leqslant l} \mathcal{J}_{l_1} \otimes \mathcal{J}_{l_2} \otimes \cdots \otimes \mathcal{J}_{l_d}. \tag{16}$$

One can define the sparse grid $\mathcal{Y}_l^d$ and approximation space U_l^d (identical to X_n) corresponding to $\mathcal{J}_l^d$ using CGL points as

$$\mathcal{Y}_l^d = \{\, \boldsymbol{x_j} : \boldsymbol{j} \in \mathcal{J}_l^d \,\}, \quad U_l^d = \{\, \varphi_{\boldsymbol{k}} : \boldsymbol{k} \in \mathcal{J}_l^d \,\}. \tag{17}$$

By letting $u_n = \sum_{\boldsymbol{k}\in J_l^d} \hat{u}_{\boldsymbol{k}} \varphi_{\boldsymbol{k}}(x)$ and $v_n = \varphi_{\boldsymbol{j}}(x), \forall \boldsymbol{j} \in J_l^d$ in equation (15), we obtain

$$\sum_{\boldsymbol{k}\in\mathcal{J}_l^d} \hat{u}_{\boldsymbol{k}} \left(\sum_{i=1}^{d} m_{k_1 j_1} \cdots m_{k_d j_d} \frac{s_{k_i j_i}}{m_{k_i j_i}} \right) + \kappa \sum_{\boldsymbol{k}\in\mathcal{J}_l^d} \hat{u}_{\boldsymbol{k}} m_{k_1 j_1} \cdots m_{k_d j_d} = \left(\mathcal{U}_l^d(f), \varphi_{\boldsymbol{j}}(x)\right), \tag{18}$$

where $m_{kj} = (\varphi_k(x), \varphi_j(x))$, $s_{kj} = (\varphi_k'(x), \varphi_j'(x))$ are the components of one-dimensional mass matrix and stiffness matrix correspondingly. Equation (18) is a symmetric linear algebraic system of size $\mathrm{Card}(\mathcal{J}_l^d)$, which can be solved efficiently by using preconditioned conjugate gradient (PCG) method, or one can explicitly build the system matrix and use other method like algebraic multigrid method or multifrontal method to solve the linear system, we refer to [18] for the efficiency comparison of different linear algebraic solvers.

2.4. Approximation error. Let u be the solution of (14), u_n be the solution of (15). In general, we have an error estimate of the form

$$\begin{aligned} \|\nabla(u-u_n)\|_{L^2} &\lesssim \inf_{v_n\in X_n}\|\nabla(u-v_n)\|_{L^2} + \sup_{\phi\in X_n;\nabla\phi\neq 0}\frac{(f,\phi)-(\mathcal{U}_l^d f,\phi)}{\|\nabla\phi\|_{L^2}} \\ &\lesssim \inf_{v_n\in X_n}\|\nabla(u-v_n)\|_{L^2} + \|f-\mathcal{U}_l^d f\|_{L^2}. \end{aligned} \tag{19}$$

Here and after "$\lesssim$" means "$\leqslant$" up to a positive constant independent of u, f and n, l.

The interpolation error on standard sparse grid $\mathcal{X}_l^d$ using CGL quadrature points was analyzed in [3]. The interpolation error estimate is:

$$\|f-\mathcal{U}_l^d(f)\|_{0,\omega} \lesssim 2^{-lk} l^{2d-1}\|f\|_{B_\omega^k} \lesssim n^{-k}(\log n)^{(k+1)(d-1)}\|f\|_{B_\omega^k}, \tag{20}$$

where $n = \mathrm{Card}(U_l^d) \approx 2^l l^{d-1}$, $\omega = \Pi_{i=1}^d (1-x_i^2)^{-1/2}$ is the d-dimensional Chebyshev weight function, and

$$B_\omega^j(I^d) = \{u : \partial_{\boldsymbol{x}}^{\boldsymbol{k}} u \in L_\omega^2(I^d),\ 0 \le |\boldsymbol{k}|_\infty \le j\}$$

is the uniformly weighted Korobov type space with norm

$$\|u\|_{B_\omega^j} = \left(\sum_{0\le|\boldsymbol{k}|_\infty\le j}\|\partial_{\boldsymbol{x}}^{\boldsymbol{k}} u\|_\omega^2\right)^{1/2}. \tag{21}$$

A projection error estimate was given by [17] for hyperbolic cross Jacobi approximation, which is asymptotically equivalent to sparse grid approximation

$$\inf_{v_n\in X_n}\|u-v_n\|_{K_{\boldsymbol{\alpha},\boldsymbol{\beta}}^j} \lesssim 2^{l(j-m)}|u|_{K_{\boldsymbol{\alpha},\boldsymbol{\beta}}^m}, \quad 0 \leqslant j \leqslant m, \tag{22}$$

where

$$K_{\boldsymbol{\alpha},\boldsymbol{\beta}}^j(I^d) = \{u : \partial_{\boldsymbol{x}}^{\boldsymbol{k}} u \in L^2_{\omega^{\boldsymbol{\alpha}+\boldsymbol{k},\boldsymbol{\beta}+\boldsymbol{k}}}(I^d),\ 0 \le |\boldsymbol{k}|_\infty \le j\} \quad \text{with}$$

$$\omega^{\boldsymbol{\alpha},\boldsymbol{\beta}} = \Pi_{i=1}^d (1-x_i)^{\alpha_i}(1+x_i)^{\beta_i}$$

is the non-uniformly weighted Korobov type space with the norm

$$\|u\|_{K_{\boldsymbol{\alpha},\boldsymbol{\beta}}^j} = \left(\sum_{0\le|\boldsymbol{k}|_\infty\le j}\|\partial_{\boldsymbol{x}}^{\boldsymbol{k}} u\|^2_{\omega^{\boldsymbol{\alpha}+\boldsymbol{k},\boldsymbol{\beta}+\boldsymbol{k}}}\right)^{1/2}, \tag{23}$$

and semi-norm

$$|u|_{K_{\boldsymbol{\alpha},\boldsymbol{\beta}}^j} = \left(\sum_{|\boldsymbol{k}|_\infty=j}\|\partial_{\boldsymbol{x}}^{\boldsymbol{k}} u\|^2_{\omega^{\boldsymbol{\alpha}+\boldsymbol{k},\boldsymbol{\beta}+\boldsymbol{k}}}\right)^{1/2}, \tag{24}$$

Combining (19), (20) and (22) with $j=1$ and $\boldsymbol{\alpha}=\boldsymbol{\beta}=-\mathbf{1}$, one gets the following error estimate:

$$\|\nabla(u_n-u)\|_{L^2} \lesssim 2^{l(1-m)}|u|_{K^m_{-\mathbf{1},-\mathbf{1}}(I^d)} + 2^{-lk}l^{2d-1}\|f\|_{B^k_\omega(I^d)}, \tag{25}$$

where $m \ge 1$ and $k > d/2$.

3. A sparse grid spectral element method using nodal bases

In this section, we construct a sparse grid spectral element method for elliptic PDEs with variable coefficients, which combines the advantage of sparse grid approximation and spectral element method. For simplicity, we shall concentrate on the two-dimensional case. The extension to higher-dimensions can be done in a similar fashion.

3.1. Galerkin formulation. We consider the following second-order elliptic PDE

$$-\nabla(\alpha(\boldsymbol{x})\nabla u(\boldsymbol{x})) + \kappa(\boldsymbol{x})u(\boldsymbol{x}) = f(\boldsymbol{x}), \quad \boldsymbol{x} \in \Omega \subset \mathbb{R}^2 \tag{26}$$

with homogeneous Dirichlet boundary condition

$$u|_{\partial\Omega} = 0 \tag{27}$$

as an example to demonstrate how to build the sparse grid spectral element method. We assume

$$\alpha(\boldsymbol{x}) \geq \alpha_0 > 0, \quad \kappa(\boldsymbol{x}) \geqslant 0, \quad \alpha(\boldsymbol{x}), \kappa(\boldsymbol{x}) \in C(\bar{\Omega}). \tag{28}$$

The weak form of the system (26)-(27) is

$$\text{Find} \quad u \in H_0^1(\Omega), \quad s.t. \quad a(u,v) = (f,v), \quad \forall \quad v \in H_0^1(\Omega), \tag{29}$$

where

$$a(u,v) = \int_\Omega \alpha(\boldsymbol{x})\nabla u \nabla v \mathrm{d}\boldsymbol{x} + \int_\Omega \kappa(\boldsymbol{x}) uv \mathrm{d}\boldsymbol{x}.$$

In the last section, we used global basis functions (polynomials, trigonometric polynomials and etc) to construct an approximation space $X_N \subset H_0^1(\Omega)$. This is very effective for smooth problems in tensor product domains. However, many interesting problems are set in non-tensor product domains, with non-smooth or singular coefficients $\alpha(\boldsymbol{x})$, $\kappa(\boldsymbol{x})$ and forcing term $f(\boldsymbol{x})$. In these situations, it is more effective to use a spectral element approach. In particular, when $\alpha(\boldsymbol{x})$, $\kappa(\boldsymbol{x})$ and $f(\boldsymbol{x})$ have known singularities, we can partition the domain Ω into elements according to the singularities.

Let $\Pi = \{\Omega_0, \ldots, \Omega_{K-1}\}$ be a quadrilateral partition of the domain Ω. On each element Ω_λ, we first map it to a unit square I^2, then use V_l^2 to approximate the mapped functions. The corresponding approximation space on the whole domain Ω in $C(\bar{\Omega})$ is denoted by V_N^Π. The corresponding interpolation points and interpolation operator are denoted by $\mathcal{X}_N^\Pi$ and $\mathcal{U}_N^\Pi$, respectively. Since the solution satisfies homogeneous Dirichlet boundary condition, the approximation space for the solution is $U_N^\Pi = V_N^\Pi \cap H_0^1(\Omega)$. The sparse grid spectral element method for equation (29) is:

$$\text{Find } u_N \in U_N^\Pi, \quad s.t. \quad a(u_N, v_N) = (\mathcal{U}_N^\Pi(f), v_N), \quad \forall \quad v_N \in U_N^\Pi. \tag{30}$$

Denote by $\mathbb{Z}_N := \{0, 1, \ldots, N-1\}$, if we choose $\{\varphi_k(\boldsymbol{x})\}_{k\in\mathbb{Z}_N}$ as a set of bases of U_N^Π, then u_N can be expressed as $u_N = \sum_{k\in\mathbb{Z}_N} \hat{u}_k \varphi_k(\boldsymbol{x})$. Plugging this into equation (30) and further taking v_N as $\varphi_j(\boldsymbol{x})$, for $j \in \mathbb{Z}_N$, we get a linear system

$$S\bar{u} + M\bar{u} = \bar{f}, \tag{31}$$

where $\bar{u} = (\hat{u}_0, \ldots, \hat{u}_{N-1})^{\mathrm{T}}$, $\bar{f} = (\hat{f}_0, ..., \hat{f}_{N-1})$, $\hat{f}_j = (\mathcal{U}_N^\Pi(f)(\boldsymbol{x}), \varphi_j(\boldsymbol{x}))$, and

$$M = (m_{jk})_{j,k\in\mathbb{Z}_N}, \qquad m_{jk} = (\kappa(\boldsymbol{x})\varphi_k(\boldsymbol{x}), \varphi_j(\boldsymbol{x})),$$
$$S = (s_{jk})_{j,k\in\mathbb{Z}_N}, \qquad s_{jk} = (\alpha(\boldsymbol{x})\nabla\varphi_k(\boldsymbol{x}), \nabla\varphi_j(\boldsymbol{x})).$$

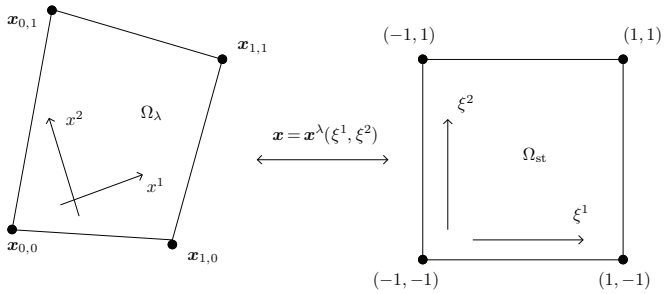


FIGURE 1. A quadrilateral element(Left) and its corresponding reference element(Right).

3.2. Quadrilateral element and mapping. We use global coordinates (x^1, x^2) for a quadrilateral element Ω_λ (see Fig. 1, Left) in physical domain, use local coordinates (ξ^1, ξ^2) in its reference domain $\Omega_{st} := (-1, 1)^2$ (see Fig. 1, Right). The mapping between local coordinates and global coordinates is a bilinear function

$$x^{i,\lambda}(\xi^1, \xi^2) = \sum_{r,s=0,1} x^{i,\lambda}_{r,s}\phi_r(\xi^1)\phi_s(\xi^2), \quad i = 1, 2, \tag{32}$$

where

$$\phi_0(\xi) = \frac{1-\xi}{2}, \quad \phi_1(\xi) = \frac{1+\xi}{2}, \tag{33}$$

and $\left\{(x^{1,\lambda}_{r,s}, x^{2,\lambda}_{r,s}) : r = 0, 1, \quad s = 0, 1\right\}$ are the coordinates of four vertices of Ω_λ. For simplicity, the parameter λ in $x^{i,\lambda}$ will be omitted when it introduces no ambiguity.

The Jacobian matrix and its determinant are given by

$$J = \left(\frac{\partial x^i}{\partial \xi^\beta}\right)_{i,\beta=1,2} = \begin{pmatrix} \partial x^1/\partial\xi^1 & \partial x^1/\partial\xi^2 \\ \partial x^2/\partial\xi^1 & \partial x^2/\partial\xi^2 \end{pmatrix}, \ \frac{\partial x^i}{\partial \xi^\beta} = \sum_{r,s=0,1} x^i_{r,s}\partial_\beta[\phi_r(\xi^1)\phi_s(\xi^2)] \tag{34}$$

and

$$|J| = \frac{\partial x^1}{\partial \xi^1}\frac{\partial x^2}{\partial \xi^2} - \frac{\partial x^1}{\partial \xi^2}\frac{\partial x^2}{\partial \xi^1}. \tag{35}$$

We have

$$J^{-1} = \begin{pmatrix} \partial\xi^1/\partial x^1 & \partial\xi^1/\partial x^2 \\ \partial\xi^2/\partial x^1 & \partial\xi^2/\partial x^2 \end{pmatrix} = \frac{1}{|J|}\begin{pmatrix} \partial x^2/\partial\xi^2 & -\partial x^1/\partial\xi^2 \\ -\partial x^2/\partial\xi^1 & \partial x^1/\partial\xi^1 \end{pmatrix}, \tag{36}$$

i.e.

$$\frac{\partial\xi^1}{\partial x^1} = \frac{1}{|J|}\frac{\partial x^2}{\partial\xi^2}, \quad \frac{\partial\xi^2}{\partial x^2} = \frac{1}{|J|}\frac{\partial x^1}{\partial\xi^1}, \quad \frac{\partial\xi^2}{\partial x^1} = -\frac{1}{|J|}\frac{\partial x^2}{\partial\xi^1}, \quad \frac{\partial\xi^1}{\partial x^2} = -\frac{1}{|J|}\frac{\partial x^1}{\partial\xi^2}. \tag{37}$$

Note that all four components in J are linear functions and $|J|$ is a bilinear function. However, components in the inverse of Jacobian matrix J^{-1} are rational functions except for the case the physical quadrilateral is a parallelogram where $|J|$ is a constant. The non-constant Jacobian determinant needs a special treatment when forming elemental mass and stiffness matrices, which will be described in next subsection.

3.3. Elemental matrices using nodal bases. Let $\{h_k(\boldsymbol{\xi})\}_{k\in\mathbb{Z}_{n_l}}$ be the Lagrange bases on sparse grid $\mathcal{X}_l^2$ in reference element Ω_{st}. i.e.

$$h_k(\boldsymbol{\xi}_j) = \delta_{kj}, \quad \forall\, \boldsymbol{\xi}_j \in \mathcal{X}_l^2, \quad k, j \in \mathbb{Z}_{n_l}, \tag{38}$$

where $n_l = \mathrm{Card}(\mathcal{X}_l^2)$ (The l in n_l is omitted when it does not introduce ambiguity). Here a special ordering algorithm is used to map the nodes in $\mathcal{X}_l^2$ into a one-dimensional vector. The bases on element Ω_λ are obtained by combining local reference bases (38) and mapping (32):

$$h_k^\lambda(\boldsymbol{x}) = \begin{cases} h_k(\boldsymbol{\xi}), & \boldsymbol{x} = (x^{1,\lambda}(\xi^1,\xi^2), x^{2,\lambda}(\xi^1,\xi^2)) \in \Omega_\lambda, \\ 0, & \boldsymbol{x} \notin \Omega_\lambda. \end{cases}$$

Then, we can form the global bases $\{\varphi_k(\boldsymbol{x}) : k \in \mathbb{Z}_N\}$ by local bases in every elements $\{h_k^\lambda(\boldsymbol{x})\}_{k\in\mathbb{Z}_n}^{\lambda\in\mathbb{Z}_K}$. Due to the continuous condition, elemental vertex (or edge) bases on adjacent elements for same vertex (or edge) point share the same global basis. The integrations in equation (30) are equivalent to the sum of integrations on every element Ω_λ.

Now let us calculate the elemental mass matrix and stiffness matrix, defined as

$$\begin{aligned} m_{jk}^\lambda &:= \int_{\Omega_\lambda} \kappa(\boldsymbol{x}) h_k^\lambda(\boldsymbol{x}) h_j^\lambda(\boldsymbol{x}) \mathrm{d}\boldsymbol{x} \\ &= \int_{\Omega_{st}} \kappa(\boldsymbol{x}(\boldsymbol{\xi}))|J(\boldsymbol{\xi})| h_k(\boldsymbol{\xi}) h_j(\boldsymbol{\xi}) \mathrm{d}\boldsymbol{\xi}, \\ s_{jk}^{\lambda,s} &:= \int_{\Omega_\lambda} \alpha(\boldsymbol{x}) \frac{\partial h_k^\lambda(\boldsymbol{x})}{\partial x^s} \frac{\partial h_j^\lambda(\boldsymbol{x})}{\partial x^s} \mathrm{d}\boldsymbol{x} \\ &= \sum_{r,r'=1,2} \int_{\Omega_{st}} \frac{\partial h_k(\boldsymbol{\xi})}{\partial \xi^r} \frac{\partial \xi^r}{\partial x^s} \alpha(x(\boldsymbol{\xi}))|J(\boldsymbol{\xi})| \frac{\partial \xi^{r'}}{\partial x^s} \frac{\partial h_j(\boldsymbol{\xi})}{\partial \xi^{r'}} d\boldsymbol{\xi}, \quad s = 1, 2, \end{aligned}$$

and $s_{jk}^\lambda = s_{jk}^{\lambda,1} + s_{jk}^{\lambda,2}$.

If $\alpha(\boldsymbol{x}), \kappa(\boldsymbol{x})$ and $|J(\boldsymbol{x})|$ are all constants, then m_{jk}^λ and $s_{jk}^{\lambda,s}$ can be evaluated efficiently with separable integrations. In general, these integrations are not separable, so we need to seek an efficient method to calculate m_{jk}^λ and $s_{jk}^{\lambda,s}$. In spectral element methods using full grids, one can use Legendre–Gauss numerical quadrature to calculate m_{jk}^λ and $s_{jk}^{\lambda,s}$. However, when we use sparse grid on each element, the corresponding numerical quadrature does not guarantee positive weights. Furthermore, the quadrature using sparse grid is not as accurate as tensor-product Gaussian quadrature. To avoid the errors with a numerical quadrature, we use a pseudo-spectral approach to calculate the components of local mass and stiff matrices. Details are given below.

Denote $L_k(\boldsymbol{\xi}) = L_{k_1}(\xi^1) L_{k_2}(\xi^2)$, $(k_1, k_2) \in \mathcal{I}_l^2$, are two-dimensional Legendre polynomials on sparse grid spectral space, with $k = k(k_1, k_2)$ being a mapping from $\mathcal{I}_l^2$ to $\mathbb{Z}_n$. According to the property of Lagrange bases, we have

$$L_k(\boldsymbol{\xi}) = \sum_{j\in\mathbb{Z}_n} L_k(\boldsymbol{\xi}_j) h_j(\boldsymbol{\xi}), \quad \forall k \in \mathbb{Z}_n.$$

Denote by $B = (L_k(\boldsymbol{\xi}_j))_{j,k\in\mathbb{Z}_n}$, $F = B^{-1} = (f_{jk})_{j,k\in\mathbb{Z}_n}$. Then $h_j(\boldsymbol{\xi}) = \sum_{j'} f_{j'j} L_{j'}(\boldsymbol{\xi})$. If $\kappa(\boldsymbol{x}(\boldsymbol{\xi}))|J(\boldsymbol{\xi})| \equiv 1$, we have

$$m_{jk}^{st} = \int_{\Omega_{st}} \sum_{k'\in\mathbb{Z}_n} f_{k'k} L_{k'}(\boldsymbol{\xi}) \sum_{j'\in\mathbb{Z}_n} f_{j'j} L_{j'}(\boldsymbol{\xi}) \mathrm{d}\boldsymbol{\xi} = \sum_{k',j'\in\mathbb{Z}_n} f_{k'k}(L_{k'}, L_{j'}) f_{j'j}. \tag{39}$$

Writing the above in matrix form, we get

$$M^{st} = F^T M_L F,$$

where M_L is the mass matrix obtained when Legendre polynomials used as bases. It is a diagonal matrix due to the orthogonality of Legendre polynomials. For non-constant coefficients or non-constant Jacobi, we use the interpolation function on sparse grid to approximate $\kappa(\boldsymbol{x}(\boldsymbol{\xi}))|J(\boldsymbol{\xi})|h_k(\boldsymbol{\xi})$, i.e.

$$\kappa(\boldsymbol{x}(\boldsymbol{\xi}))|J(\boldsymbol{\xi})|h_k(\boldsymbol{\xi}) \approx \mathcal{U}_l^2(\kappa(\boldsymbol{x}(\boldsymbol{\xi}))|J(\boldsymbol{\xi})|h_k(\boldsymbol{\xi})) = \kappa(\boldsymbol{x}(\boldsymbol{\xi}_k))|J(\boldsymbol{\xi}_k)|h_k(\boldsymbol{\xi}), \tag{40}$$

where $\mathcal{U}_l^2$ is the interpolation operator on sparse grid $\mathcal{X}_l^2$. Thus

$$\begin{aligned} m_{jk}^{\lambda} &\approx \int_{\Omega_{st}} \kappa(\boldsymbol{x}(\boldsymbol{\xi}_k))|J(\boldsymbol{\xi}_k)|h_k(\boldsymbol{\xi})h_j(\boldsymbol{\xi})\mathrm{d}\boldsymbol{\xi} \\ &= |J(\boldsymbol{\xi}_k)|b(\boldsymbol{x}(\boldsymbol{\xi}_k)) \sum_{k',j'} f_{k'k}(L_{k'}, L_{j'}) f_{j'j}. \end{aligned}$$

Written in matrix form, we have

$$M^{\lambda} \approx F^T M_L F \cdot \Lambda(b|J|), \tag{41}$$

where $\Lambda(f(\cdot))$ denote the diagonal matrix formed by the values of function $f(\cdot)$ at sparse grid points $\mathcal{X}_l^2$.

For the stiffness matrix, we need to use the derivative matrix. Let $D_r = (d_{kj}^r)_{j,k\in\mathbb{Z}_n}$, $r = 1, 2$ be the first order differentiation matrix w.r.t. ξ^r, then

$$\frac{\partial h_k(\boldsymbol{\xi})}{\partial \xi^r} = \sum_{j\in\mathbb{Z}_n} d_{kj}^r h_j(\boldsymbol{\xi}).$$

Here $\{\partial h_k/\partial \xi^r,\ k \in \mathbb{Z}_n\}$ need to be contained in $\mathrm{span}\{h_j, j \in \mathbb{Z}_n\}$. This is not a problem for sparse grid elements including boundary points $\mathcal{X}_l^d$, but is not satisfied by elements with less edge points, which will be introduced in the next section with a special treatment.

Denote by

$$\mu_{r,r'}^s(\boldsymbol{\xi}) = \alpha(\boldsymbol{x}(\boldsymbol{\xi}))|J(\boldsymbol{\xi})| \frac{\partial \xi^r}{\partial x^s} \frac{\partial \xi^{r'}}{\partial x^s},$$

we have

$$\begin{aligned}
s_{jk}^{\lambda,s} &= \int_{\Omega_{st}} \sum_{r,r'=1,2} \frac{\partial h_k(\boldsymbol{\xi})}{\partial \xi^r} \frac{\partial \xi^r}{\partial x^s} \alpha(\boldsymbol{x}(\boldsymbol{\xi}))|J(\boldsymbol{\xi})| \frac{\partial \xi^{r'}}{\partial x^s} \frac{\partial h_j(\boldsymbol{\xi})}{\partial \xi^{r'}} d\boldsymbol{\xi} \\
&= \sum_{r,r'=1,2} \int_{\Omega_{st}} \frac{\partial h_k(\boldsymbol{\xi})}{\partial \xi^r} \mu_{r,r'}^s(\boldsymbol{\xi}) \frac{\partial h_j(\boldsymbol{\xi})}{\partial \xi^{r'}} d\boldsymbol{\xi} \\
&= \sum_{r,r'=1,2} \int_{\Omega_{st}} \left(\sum_{k' \in \mathbb{Z}_n} d_{kk'}^r h_{k'}(\boldsymbol{\xi}) \right) \mu_{r,r'}^s(\boldsymbol{\xi}) \left(\sum_{j' \in \mathbb{Z}_n} d_{jj'}^{r'} h_{j'}(\boldsymbol{\xi}) \right) d\boldsymbol{\xi} \\
&= \sum_{r,r'=1,2} \sum_{k',j' \in \mathbb{Z}_n} d_{kk'}^r d_{jj'}^{r'} \int_{\Omega_{st}} \mu_{r,r'}^s(\boldsymbol{\xi}) h_{k'}(\boldsymbol{\xi}) h_{j'}(\boldsymbol{\xi}) d\boldsymbol{\xi} \\
&\approx \sum_{r,r'=1,2} \sum_{k',j',\alpha,\beta} d_{kk'}^r [\mu_{r,r'}^s(\boldsymbol{\xi}_{k'}) f_{\beta k'}(L_\beta, L_\alpha) f_{\alpha j'}] d_{jj'}^{r'},
\end{aligned}$$

which can be written in matrix form

$$S^{\lambda,s} \approx \sum_{r,r'=1,2} D_r F^T M_L F \Lambda(\mu_{r,r'}^s) D_{r'}^T. \tag{42}$$

3.4. Assembling of matrices. For a given function $u_N(\boldsymbol{x}) \in U_N^{\Pi}$, we have global expansion $u_N(\boldsymbol{x}) = \sum_{k \in \mathbb{Z}_N} \hat{u}_k \varphi_k(\boldsymbol{x})$ and local expansion $u_N(\boldsymbol{x})|_{\Omega_\lambda} = \sum_{k \in \mathbb{Z}_n} u_k^\lambda h_k^\lambda(\boldsymbol{x})$. $\{h_k^\lambda, k \in \mathbb{Z}_n\}$ are ordered such that $h_1^\lambda, h_2^\lambda, h_3^\lambda, h_4^\lambda$ are the bases corresponding to four vertices, $h_1^\lambda, \ldots, h_{4p}^\lambda$ are the bases corresponding to the $4p$ edge points with $p = m_l$, the other bases have nonzero values only on inner quadrature points. Denote by $\bar{u}_\lambda$ the spectral expansion coefficient of $u_N(\boldsymbol{x})$ on element Ω_λ, i.e. $\bar{u}_\lambda = (u_0^\lambda, ..., u_{n-1}^\lambda)^T$. Let $\underline{u} = (\bar{u}_0^T, ..., \bar{u}_{K-1}^T)^T$ be the collection of all local expansion coefficients and $\bar{u} = (\hat{u}_0, ..., \hat{u}_{N-1})^T$ the global expansion of $u_N(\boldsymbol{x})$. According to the continuous condition of $u_N(\boldsymbol{x})$, expansion coefficients in two adjacent elements corresponding to the same edge or vertex point are referred to the same global coefficient, which means $N \leqslant nK$. We use a degree of freedom (DoF) mapping function $\theta(\cdot,\cdot) : \mathbb{Z}_n \times \mathbb{Z}_K \to \mathbb{Z}_N$ to map the local DoF index to global index, where $\theta(k,\lambda) = \alpha$ means u_k^λ and $\hat{u}_\alpha$ are the coefficients for the same point, thus equal to each other. Define scattering matrix $P = (p_{ij})_{i,j} \in (\mathbb{Z}_2)^{Kn \times N}$, where

$$p_{ij} = \begin{cases} 1, & \text{if } \theta\left(\mathrm{mod}(i,n), \lfloor \frac{i}{n} \rfloor\right) = j, \\ 0, & \text{otherwise.} \end{cases}$$

Then $\underline{u} = P\bar{u}$. If we define $P_\lambda = (p_{ij})_{i=n\lambda,\ldots,n\lambda+n-1}^{j=0,\ldots,N-1}$, then $\bar{u}_\lambda = P_\lambda \bar{u}$. It is easy to check that the global mass and stiffness matrices are given by

$$M = P^T \underline{M} P, \quad S = P^T \underline{S} P,$$

where $\underline{M} = \mathrm{diag}(M^0, ..., M^K)$, $\underline{S} = \mathrm{diag}(S^0, ..., S^K)$ are block diagonal matrix with block diagonals being the corresponding elemental mass and stiffness matrices. We denote the overall system matrix $M + S$ by A, so the equation (31) can be written as

$$A\bar{u} = \bar{f}. \tag{43}$$

3.5. Solving the linear system. The system matrix A has a special block structure

$$A = \begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix},$$

where A_{22} is a block diagonal matrix with K block diagonals matrices corresponding to the interaction among inner DoFs in K elements, so it can be inverted efficiently. A_{11} corresponds to the interaction between vertex DoFs and edge DoFs. A_{12} and A_{21} are interactions between vertex/edge DoFs and inner DoFs. The linear system (43) can be written as

$$\begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix} \begin{pmatrix} \bar{u}_1 \\ \bar{u}_2 \end{pmatrix} = \begin{pmatrix} \bar{f}_1 \\ \bar{f}_2 \end{pmatrix},$$

This system can be solved in two steps using a Schur-complement approach

$$(A_{11} - A_{12}A_{22}^{-1}A_{21})\bar{u}_1 = \bar{f}_1 - A_{12}A_{22}^{-1}\bar{f}_2, \tag{44}$$

$$\bar{u}_2 = A_{22}^{-1}\bar{f}_2 - A_{22}^{-1}A_{21}\bar{u}_1. \tag{45}$$

The sparsity patterns of system matrix A and its Schur-complement $A_{11}-A_{12}A_{22}^{-1}A_{21}$ are given in Figure 6:(a),(b). Equation (44) is a much smaller system than (43), which can be solved by direct methods or PCG methods. A simple and effective choice for the preconditioner is the block diagonal preconditioner with two blocks, one consists of all vertex DoFs, the other one consists of all edge DoFs (see [14] and references therein).

4. Sparse grid element with less edge degrees of freedom

It is known that the condition number of the Schur-complement of a spectral element method depends on the number of DoFs on edges, and the distribution of the nodes on edges. In our sparse grid spectral element method, we can let the degree of freedoms on edges be a free parameter to make the method more flexible. In the 2-d case, for each quadrilateral element, we have four vertex degrees of freedom. For each edge, we have $m+1$ degrees of freedom including two boundary points, for the inner part of the element, we have $n-4m$ degree of freedoms, adding up account for n degree of freedoms in this element. Since we use polynomial approximation, the corresponding approximation space for the reference element Ω_{st} is

$$W_{l,m}^2 = \left\{ u \in V_l^2 : \quad u|_{E_i} = P_m(E_i), i = 1,2,3,4 \right\},$$

where $E_i, i = 1,2,3,4$ are four edges of the reference domain Ω_{st}. In this section, for simplicity, we use $\boldsymbol{x} = (x, y)$ as the 2-dimensional coordinates in the reference domain Ω_{st}. So $E_1 = \{\boldsymbol{x} \in \Omega_{st} : y = -1\}$, $E_2 = \{\boldsymbol{x} \in \Omega_{st} : y = 1\}$, $E_3 = \{\boldsymbol{x} \in \Omega_{st} : x = -1\}$, $E_4 = \{\boldsymbol{x} \in \Omega_{st} : x = 1\}$. From the definition, we see that V_l^2 has the following inner-outer decomposition

$$V_l^2 = \operatorname{span}\{u_i(x)\phi_i(y),\ \phi_i(x)v_i(y),\ u_i, v_i \in P_{m_l-1},\ i \in \mathbb{Z}_2\} \oplus U_l^2,$$

where $E = \cup_{i=1}^4 E_i$. From the above identity, we know that $m \leqslant m_l - 1$, and $W_{l,m}^2$ can be written as

$$W_{l,m}^2 = \operatorname{span}\{u_i(x)\phi_i(y),\ \phi_i(x)v_i(y),\ u_i, v_i \in P_m,\ i \in \mathbb{Z}_2\} \oplus U_l^2.$$

When $m = m_l - 1$, $W_{l,m}^2 = V_l^2$ it is a natural interpolation space on sparse grids $\mathcal{X}_l^2$. One can calculate the Lagrange interpolation polynomial by fast spectral transform on sparse grid, and derivatives of any function in space V_l^2 is still in V_l^2.

When $m < m_l - 1$, the space $W_{l,m}^2$ is not closed with respect to differentiation. For example, $u(\boldsymbol{x}) = (L_k(x) - L_{k+2}(x))(L_0(y) - L_2(y)) \in W_{l,m}^2$ for $m < k + 2 \leqslant m_l - 1$, but $\partial_y u = (L_k(x) - L_{k+2}(x))L_1(y)$, it does not belongs to P_m on E_1, E_2. This complicates the implementation of differentiation on sparse grid element and the procedure to calculate the stiffness matrix.

To overcome this difficulty, we enlarge the space $W_{l,m}^2$ to V_l^2 whenever we want to take differentiation or fast transform. After all the calculation on V_l^2, we project the function back to $W_{l,m}^2$ as described below. In the inner region, two different interpolations use same points, and their corresponding Lagrange interpolation polynomials are identical, since they all vanish on $\partial\Omega_{st}$. The bases need to be projected are those on the four edges and vertices.

4.1. Edge Lagrange bases mapping. We take the first edge as an example. The treatment for other three edges are similar. We first introduce some notations. Let

$$\{x_0 = -1, \quad x_1 = 1, \quad x_2 = 0, \quad x_3 = -\sqrt{2}/2, \quad x_4 = \sqrt{2}/2, \quad x_5 = \ldots, \quad \ldots\}$$

be the one-dimensional hierarchical CGL points used in both x- and y-direction to construct $\mathcal{X}_l^2$. Let $\{x_i^W, i = 0, \ldots, m\}$ with $x_0^W = -1$, $x_1^W = 1$ be the quadrature points used for four edges in forming $W_{l,m}^2$, $\mathcal{K}_{l,m}^2 := \mathcal{J}_l^2 \cup \mathcal{B}_m^2$ be the basis index set of $W_{l,m}^2$. Here

$$\mathcal{B}_m^2 := \{(i_1, i_2) : i_1 = \mathbb{Z}_{m+1}, \quad i_2 = 0, 1\} \cup \{(i_1, i_2) : i_1 = 0, 1; i_2 \in \mathbb{Z}_{m+1}\}.$$

Then $\mathcal{Z}_{l,m}^2$, the interpolation points for $W_{l,m}^2$ is given by

$$\mathcal{Z}_{l,m}^2 = \mathcal{Y}_l^2 \cup \{\boldsymbol{x}_{\boldsymbol{j}}^W := (x_{j_1}^W, x_{j_2}^W) \,|\, \boldsymbol{j} \in \mathcal{B}_m^2\}.$$

Denote by $\mathcal{I}_V = \{\boldsymbol{j} : j_1 = 0, 1; j_2 = 0, 1\}$ the index set of four vertices, $\mathcal{I}_F = \mathcal{J}_l^2$ be the index set of all inner nodes, $\mathcal{I}_{E_i}^V$ (corresp $\mathcal{I}_{E_i}^W$) be subset of $\mathcal{I}_l^2$ (corresp $\mathcal{Z}_{l,m}^2$) correspond to points on E_i including two vertices. Let $\mathcal{I}_{E_i^0}^V$ and $\mathcal{I}_{E_i^0}^W$ be subset of $\mathcal{I}_l^2$ and $\mathcal{Z}_{l,m}^2$ corresponding to points on E_i excluding the vertices. Denote $\mathcal{I}_E^W = \cup_{i=1}^4 \mathcal{I}_{E_i}^W$, $\mathcal{I}_E^V = \cup_{i=1}^4 \mathcal{I}_{E_i}^V$. Figure 2 shows the interpolation points of V_3^2 (left) and $W_{3,2}^2$ (right).

Let $h_{\boldsymbol{k}}^W(\boldsymbol{x})$,$\boldsymbol{k} \in \mathcal{K}_{l,m}^2$ be the Lagrange bases corresponding to $\boldsymbol{x}_{\boldsymbol{k}} \in \mathcal{Z}_{l,m}^2$, $h_{\boldsymbol{k}}^V(\boldsymbol{x})$, $\boldsymbol{k} \in \mathcal{I}_l^2$ be the Lagrange bases corresponding to $\boldsymbol{x}_{\boldsymbol{k}} \in \mathcal{X}_l^2$. By the definition of Lagrange bases, we have

$$h_{\boldsymbol{k}}^V(\boldsymbol{x}_{\boldsymbol{j}}) = \delta_{\boldsymbol{k}\boldsymbol{j}}, \quad \boldsymbol{k}, \boldsymbol{j} = \mathcal{I}_{E_1}^V$$

and

$$h_{\boldsymbol{k}}^W(\boldsymbol{x}_{\boldsymbol{j}}) = \delta_{\boldsymbol{k}\boldsymbol{j}}, \quad \boldsymbol{k}, \boldsymbol{j} = \mathcal{I}_{E_1}^W.$$

Proposition 1. *The basis functions $h_{\boldsymbol{k}}^W, \boldsymbol{k} \in \mathcal{I}_{E_1^0}^W$ are related to $h_{\boldsymbol{j}}^V, \boldsymbol{j} \in \mathcal{I}_{E_1^0}^V$ through the following formula*

$$h_{\boldsymbol{k}}^W(\boldsymbol{x}) = \sum_{\boldsymbol{j} \in \mathcal{I}_{E_1^0}^V} c_{k_1 j_1} h_{\boldsymbol{j}}^V(\boldsymbol{x}), \quad \boldsymbol{k} \in \mathcal{I}_{E_1^0}^W \tag{46}$$

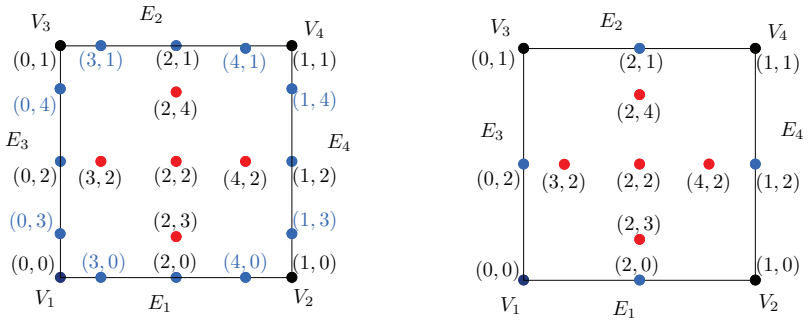


FIGURE 2. The distribution of the interpolation points of $\mathcal{X}_3^2$(Left, $m_l = 5$) and $\mathcal{Z}_{3,2}^2$ (Right, $m+1 = 3$).

where

$$c_{kj} = \ell_k^{m+1}(x_j). \tag{47}$$

Here $\ell_k^{m+1}(x)$ *is the 1-d Lagrange bases on interpolation set* $\{x_j^W, j = 0, \dots, m\}$ *corresponding to* x_k^W*. If* $x_k^W = x_k$*, for* $k = 0, \dots, m$*, then*

$$C := (c_{ij})_{i=0,\dots,m}^{j=0,\dots,m_l-1} = (I_{m+1}, C_2),$$

where I_{m+1} *is a* $(m+1) \times (m+1)$ *identity matrix.* $C_2 = (c_{ij})_{i=0,\dots,m}^{j=m+1,\dots,m_l-1}$.

Proof. For bases corresponding to DoFs on edge E_1 but not a vertex, we have the following formula

$$h_{\boldsymbol{j}}^V(\boldsymbol{x}) = \ell_{j_1}^{m_l}(x)\varphi_0(y) - \sum_{\boldsymbol{\gamma}\in\mathcal{I}_F} \ell_{j_1}^{m_l}(x_{\gamma_1})\varphi_0(y_{\gamma_2})h_{\boldsymbol{\gamma}}^V(\boldsymbol{x}), \quad \boldsymbol{j} \in \mathcal{I}_{E_1^0}^V, \tag{48}$$

$$h_{\boldsymbol{k}}^W(\boldsymbol{x}) = \ell_{k_1}^{m+1}(x)\varphi_0(y) - \sum_{\boldsymbol{\gamma}\in\mathcal{I}_F} \ell_{k_1}^{m+1}(x_{\gamma_1})\varphi_0(y_{\gamma_2})h_{\boldsymbol{\gamma}}^W(\boldsymbol{x}), \quad \boldsymbol{k} \in \mathcal{I}_{E_1^0}^W, \tag{49}$$

where $\ell_j^{m_l}(x)$ is the Lagrange basis of edge points of $\mathcal{X}_l^2$ corresponding to x_j. Plugging the above two expressions into (46), and noticing that $h_{\boldsymbol{\gamma}}^V = h_{\boldsymbol{\gamma}}^W$, for all $\boldsymbol{\gamma} \in \mathcal{I}_F$, the equality (46) holds if and only if

$$\ell_{k_1}^{m+1}(x) = \sum_{\boldsymbol{j}\in\mathcal{I}_{E_1^0}^V} c_{k_1 j_1}\ell_{j_1}^{m_l}(x), \tag{50}$$

Setting $x = x_{\gamma_1}$, for all $\boldsymbol{\gamma} \in \mathcal{I}_{E_1^0}^V$, we have

$$\ell_{k_1}^{m+1}(x_{\gamma_1}) = \sum_{\boldsymbol{j}\in\mathcal{I}_{E_1^0}^V} c_{k_1 j_1}\delta_{j_1\gamma_1} = c_{k_1\gamma_1}, \quad \text{for } \boldsymbol{k} \in \mathcal{I}_{E_1^0}^W, \quad \boldsymbol{\gamma} \in \mathcal{I}_{E_1^0}^V. \tag{51}$$

Thus, the identity (46) holds by the definition c_{kj} and (47). □

There are two ways to calculate c_{kj}. One is to use the definition of 1-d Lagrange interpolate:

$$\ell_k^{m+1}(x_j) = \frac{\Pi_{i=0,i\neq j}^m (x_j - x_i^W)}{\Pi_{i=0,i\neq j}^m (x_j^W - x_i^W)}. \tag{52}$$

The computational cost for this approach is $O(m^2 m_l)$. Another approach is to use an orthogonal transform, i.e. from

$$L_k(x) = \sum_{j=0}^{m} L_k(x_j)\ell_j^{m+1}(x), \quad k = 0, \ldots, m.$$

Denote $B = (L_k(x_j))_{k,j=0,\ldots,m}$, $F = (f_{kj}) = B^{-1}$. Then

$$\ell_k^{m+1}(x_j) = \sum_{n=0}^{m} f_{kn} L_n(x_j) \tag{53}$$

Using this approach, one need to evaluate $(L_k(x_j))_{j=0,\ldots,m_l-1}^{k=0,\ldots,m}$, which can be pre-computed, then invert B and do a matrix-matrix product. The computational cost is also $O(m^2 m_l)$.

4.2. Vertex Lagrange bases mapping. We take the first vertex bases as an example. According to the ordering of the geometry, it lies on both E_1 and E_3. The Lagrange bases for W is

$$h_{(0,0)}^W(\boldsymbol{x}) = \ell_0^{m+1}(x)\varphi_0(y) - \sum_{\boldsymbol{j}\in\mathcal{I}_{\tilde{E}_1}^W} \ell_0^{m+1}(x_{j_1})\varphi_0(y_{j_2}) h_{\boldsymbol{j}}^W(\boldsymbol{x}), \tag{54}$$

where $\mathcal{I}_{\tilde{E}_1}^W = \mathcal{I}_{E_3^0}^W \cup \mathcal{I}_{E_4^0}^W \cup \mathcal{I}_F$. To related this bases to the Lagrange bases for V, we need to use bases

$$h_{\boldsymbol{j}}^V(\boldsymbol{x}) = \ell_{j_1}^{m_l}(x)\varphi_0(y) - \sum_{\boldsymbol{\gamma}\in\mathcal{I}_{\tilde{E}_1}^V} \ell_{j_1}^{m_l}(x_{\gamma_1})\varphi_0(y_{\gamma_2}) h_{\boldsymbol{\gamma}}^V(\boldsymbol{x}), \quad \boldsymbol{j} \in \mathcal{I}_{E_1}^V, \tag{55}$$

and the identity

$$\ell_0^{m+1}(x) = \sum_{\boldsymbol{j}\in\mathcal{I}_{E_1^V}} c_{0j_1} \ell_{j_1}^{m_l}(x). \tag{56}$$

Using equation (54), (55), (56), we get

$$\begin{aligned}
&h_{(0,0)}^W(\boldsymbol{x}) - \sum_{\boldsymbol{j}\in\mathcal{I}_{E_1^V}} c_{0j_1} h_{\boldsymbol{j}}^V(\boldsymbol{x}) \\
&= \sum_{\boldsymbol{j}\in\mathcal{I}_{E_1}^V} c_{0j_1} \Bigg(\sum_{\boldsymbol{\gamma}\in\mathcal{I}_{\tilde{E}_1}^V} \ell_{j_1}^{m_l}(x_{\gamma_1})\varphi_0(y_{\gamma_2}) h_{\boldsymbol{\gamma}}^V(\boldsymbol{x}) \Bigg) - \sum_{\boldsymbol{j}\in\mathcal{I}_{\tilde{E}_1}^W} \ell_0^{m+1}(x_{j_1})\varphi_0(y_{j_2}) h_{\boldsymbol{j}}^W(\boldsymbol{x}) \\
&= \sum_{\boldsymbol{\gamma}\in\mathcal{I}_{\tilde{E}_1}^V} \ell_0^{m+1}(x_{\gamma_1})\varphi_0(y_{\gamma_2}) h_{\boldsymbol{\gamma}}^V(\boldsymbol{x}) - \sum_{\boldsymbol{j}\in\mathcal{I}_{\tilde{E}_1}^W} \ell_0^{m+1}(x_{j_1})\varphi_0(y_{j_2}) h_{\boldsymbol{j}}^W(\boldsymbol{x}) \\
&= \sum_{\boldsymbol{j}\in\mathcal{I}_{E_3^0}^V} \ell_0^{m+1}(x_{j_1})\varphi_0(y_{j_2}) h_{\boldsymbol{j}}^V(\boldsymbol{x}) - \sum_{\boldsymbol{j}\in\mathcal{I}_{E_3^0}^W} \ell_0^{m+1}(x_{j_1})\varphi_0(y_{j_2}) h_{\boldsymbol{j}}^W(\boldsymbol{x}) \\
&= \sum_{\boldsymbol{j}\in\mathcal{I}_{E_3^0}^V} \varphi_0(y_{j_2}) h_{\boldsymbol{j}}^V(\boldsymbol{x}) - \sum_{\boldsymbol{j}\in\mathcal{I}_{E_3^0}^W} \varphi_0(y_{j_2}) h_{\boldsymbol{j}}^W(\boldsymbol{x}).
\end{aligned} \tag{57}$$

Then for $h^W_{\boldsymbol{j}}(\boldsymbol{x}), \boldsymbol{j} \in \mathcal{I}^W_{E^0_3}$, using Proposition 1, we have

$$h^W_{\boldsymbol{j}}(\boldsymbol{x}) = \sum_{\boldsymbol{k}\in\mathcal{I}^V_{E^0_3}} c_{j_2k_2} h^V_{\boldsymbol{k}}(\boldsymbol{x}), \quad \boldsymbol{k} \in \mathcal{I}^W_{E^0_3}.$$

Plugging it in (57), we get

$$\begin{aligned} h^W_{(0,0)}(\boldsymbol{x}) &= \sum_{\boldsymbol{j}\in\mathcal{I}^V_{E_1}} c_{0j_1} h^V_{\boldsymbol{j}}(\boldsymbol{x}) + \sum_{\boldsymbol{j}\in\mathcal{I}^V_{E^0_3}} \varphi_0(y_{j_2}) h^V_{\boldsymbol{j}}(\boldsymbol{x}) - \sum_{\boldsymbol{j}\in\mathcal{I}^W_{E^0_3}} \varphi_0(y_{j_2}) \sum_{\boldsymbol{k}\in\mathcal{I}^V_{E^0_3}} c_{j_2k_2} h^V_{\boldsymbol{k}}(\boldsymbol{x}) \\ &= \sum_{\boldsymbol{j}\in\mathcal{I}^V_{E_1}} c_{0j_1} h^V_{\boldsymbol{j}}(\boldsymbol{x}) + \sum_{\boldsymbol{j}\in\mathcal{I}^V_{E^0_3}} \left[\varphi_0(y_{j_2}) - \sum_{\boldsymbol{k}\in\mathcal{I}^W_{E^0_3}} \varphi_0(y_{k_2}) c_{k_2j_2}\right] h^V_{\boldsymbol{j}}(\boldsymbol{x}) \\ &= h^V_{(0,0)}(\boldsymbol{x}) + \sum_{\boldsymbol{j}\in\mathcal{I}^V_{E^0_1}} c_{0j_1} h^V_{\boldsymbol{j}}(\boldsymbol{x}) + \sum_{\boldsymbol{j}\in\mathcal{I}^V_{E^0_3}} \left[\varphi_0(y_{j_2}) - \sum_{\boldsymbol{k}\in\mathcal{I}^W_{E^0_3}} \varphi_0(y_{k_2}) c_{k_2j_2}\right] h^V_{\boldsymbol{j}}(\boldsymbol{x}). \end{aligned}$$

Since

$$\varphi_0(y) = \sum_{\boldsymbol{k}\in\mathcal{I}^W_{E_3}} \varphi_0(y_{k_2}) \ell^{m+1}_{k_2}(y),$$

taking $y = y_{j_2}$, we get $\varphi_0(y_{j_2}) - \sum_{\boldsymbol{k}\in\mathcal{I}^W_{E^0_3}} \varphi_0(y_{k_2}) c_{k_2j_2} = \varphi_0(y_0) c_{0j_2} = c_{0j_2}$. So, we have

$$h^W_{(0,0)}(\boldsymbol{x}) = h^V_{(0,0)}(\boldsymbol{x}) + \sum_{\boldsymbol{j}\in\mathcal{I}^V_{E^0_1}} c_{0j_1} h^V_{\boldsymbol{j}}(\boldsymbol{x}) + \sum_{\boldsymbol{j}\in\mathcal{I}^V_{E^0_3}} c_{0j_2} h^V_{\boldsymbol{j}}(\boldsymbol{x}).$$

Using a similar treatment to other three vertices, we can derive the following result:

Proposition 2. *The four vertex Lagrange bases $h^W_{\boldsymbol{j}}(\boldsymbol{x}), \boldsymbol{j} \in \mathcal{I}_V$ can be written as linear combinations of Lagrange bases $\{h^V_{\boldsymbol{k}}(\boldsymbol{x}), \quad \boldsymbol{k} \in \mathcal{I}^V_E\}$ as*

$$h^W_{\boldsymbol{j}}(\boldsymbol{x}) = h^V_{\boldsymbol{j}}(\boldsymbol{x}) + \sum_{\boldsymbol{k}\in\mathcal{I}^V_{E^x_{\boldsymbol{j}}}} c_{j_1k_1} h^V_{\boldsymbol{k}}(\boldsymbol{x}) + \sum_{\boldsymbol{k}\in\mathcal{I}^V_{E^y_{\boldsymbol{j}}}} c_{j_2k_2} h^V_{\boldsymbol{k}}(\boldsymbol{x}), \quad \boldsymbol{j} \in \mathcal{I}_V, \tag{58}$$

where $E^x_{\boldsymbol{j}} = E^0_{j_2+1}$, $E^y_{\boldsymbol{j}} = E^0_{j_1+3}$.

Using Propositions 1 and 2, we can now form elemental mass and stiffness matrices for bases $\{h^W_{\boldsymbol{j}}, \boldsymbol{j} \in W^2_{l,m}\}$.

5. Numerical results

We start with some comments on the error estimates.

Let u be the solution of (29), u_N be the solution of (30). By the first Strang's lemma, we have an error estimate of the form

(59)

$$\begin{aligned} \|\nabla(u-u_N)\|_{L^2(\Omega)} &\lesssim \inf_{v_N\in U^\Pi_N} \|\nabla(u-v_N)\|_{L^2(\Omega)} + \sup_{\phi\in U^\Pi_N;\nabla\phi\neq 0} \frac{(f,\phi) - (\mathcal{U}^\Pi_N f,\phi)}{\|\nabla\phi\|_{L^2(\Omega)}} \\ &\lesssim \inf_{v_N\in U^\Pi_N} \|\nabla(u-v_N)\|_{L^2(\Omega)} + \|f - \mathcal{U}^\Pi_N f\|_{L^2(\Omega)}. \end{aligned}$$

Hence, we just need to estimate the error of the projection $H^1_0(\Omega) \to U^\Pi_N$ in $H^1(\Omega)$, and the error of interpolation $C(\Omega) \to V^\Pi_N$ in $L^2(\Omega)$. For one-element spectral sparse grid method, these are given in Section 2. For multi-element spectral sparse

grid method, we need to assume that the partition is quasi-uniform, i.e., the lengths of four edges and four angles of each quadrilateral in the partition have a lower and upper bound, such that the Jacobian on each element satisfies

$$ch^2 \le |J^\lambda(\boldsymbol{x})| \le Ch^2, \quad \forall\ \lambda = 0, \ldots, K-1, \tag{60}$$

where h is the maximum length of the edges, and c, C are two positive real number. Then, by using a rather standard but very tedious procedure in the analysis of spectral-element method [8, 12], one can derive a detailed description of the above projection and interpolation errors in (59). We omit the detail for the sake of brevity.

We use several examples solving equation (26) to test the accuracy and efficiency of the proposed sparse grid spectral element methods. To present the results, we use abbreviation sgSEM-N for the sparse grid spectral element method using nodal bases constructed in Section 3, sgSEM-Nm for the sparse grid spectral element method with less DoFs on edges constructed in Section 4. In sgSEM-N, we can generate a full Chebyshev–Gauss–Lobatto grid by letting total level number $l = 0$ and varying m_0, the corresponding results are denoted by sgSEM-N-F.

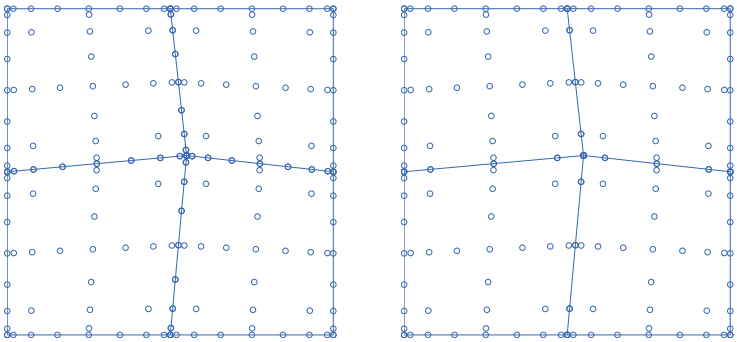


FIGURE 3. A four-element domain partition with different interpolation points in first numerical example. Left: Chebyshev–Gauss–Lobatto sparse grids $\mathcal{X}_4^2$; Right: Chebyshev–Gauss–Lobatto sparse grids with less edge DoFs $\mathcal{Z}_{4,4}^2$.

Example 1. $\alpha(\boldsymbol{x}) \equiv (x_1 + x_2 + 2)^2$, $\kappa(\boldsymbol{x}) \equiv (x_1 + x_2 + 4)^2$, $f(\boldsymbol{x})$ is chosen such that the exact solution is

$$u_1(\boldsymbol{x}) = \sin\left(\pi \frac{x_1 + 1}{2}\right) \sin\left(\pi \frac{x_2 + 1}{2}\right). \tag{61}$$

The domain is $[-1, 1]^2$, and four non-uniform elements are used (see Figure 3).

Example 2. $\alpha(\boldsymbol{x}) \equiv 1$, $\kappa(\boldsymbol{x}) \equiv 1$, $f(\boldsymbol{x}) = 1$, and the exact solution is given by

$$u_2(\boldsymbol{x}) = \sum_{k_1,k_2 \text{ odd}} \frac{16}{k_1 k_2 \pi^2} \frac{4}{4b + k_1^2\pi^2 + k_2^2\pi^2} \sin\left(k_1\pi \frac{x_1 + 1}{2}\right) \sin\left(k_2\pi \frac{x_2 + 1}{2}\right). \tag{62}$$

The domain is $[-1, 1]^2$, and sixteen uniform square elements are used.

Example 3. $\alpha(\boldsymbol{x}) = 1, \kappa(\boldsymbol{x}) = 1$, $f(\boldsymbol{x})$ is chosen such that the exact solution is

$$u_3(x_1, x_2) = (1 - [x_1]^2)(1 - [x_2]^2)[e^{-|x_1|^3 - |x_2|^3} + e^{-|x_1 - 0.5|^5 - |x_2 - 0.5|^5}]. \tag{63}$$

The domain is $[-1, 1]^2$. Both one element and uniform four-square elements are used.

Example 4. $\alpha(\boldsymbol{x}) = 1, \kappa(\boldsymbol{x}) = 1$, $f(\boldsymbol{x})$ is chosen such that the exact solution is

$$u_4(x_1, x_2) = (x_1^2 - x_1^4)(x_2^2 - x_2^4)[e^{-|x_1|^3 - |x_2|^3} + e^{-|x_1 + 0.5|^5 - |x_2 - 0.5|^5}]. \tag{64}$$

The domain is L-shaped as shown in Figure 5:(a). Three square elements are used for it.

TABLE 1. The L^2 errors of sparse grid spectral element methods sgSEM-N-F, sgSEM-N, sgSEM-Nm for Example 1. The number of edge nodes for sgSEM-Nm is $m + 1 = m_{l-1}$.

sgSEM-N-F				sgSEM-N				sgSEM-Nm			
l	m_0	DoFs	L^2 error	l	m_0	DoFs	L^2 error	l	m_0	DoFs	L^2 error
0	3	9	1.23E-2	0	3	9	1.23E-2	0	3	9	
0	5	49	3.23E-4	1	3	33	1.27E-3	1	3	25	1.10E-2
0	7	121	2.65E-6	2	3	97	9.43E-5	2	3	81	1.25E-4
0	9	225	1.15E-8	1	5	161	2.02E-6	1	5	145	8.95E-5
0	11	361	3.29E-11	3	3	257	2.08E-6	3	3	225	2.08E-6
0	13	529	6.76E-14	1	7	385	4.33E-10	1	7	361	5.17E-7
0	15	729	6.25E-15	4	3	641	3.21E-10	4	3	577	3.21E-10

TABLE 2. The convergence results of sgSEM-N-F and sgSEM-Nm using 16 elements for Example 2 with corner singularity. The number of edge nodes for sgSEM-Nm is $m + 1 = m_{l-1}$.

sgSEM-N-F				sgSEM-N				sgSEM-Nm			
l	m_0	DoFs	L^2 error	l	m_0	DoFs	L^2 error	l	m_0	DoFs	L^2 error
0	3	49	2.29E-4	0	3	49	2.29E-4				
0	7	529	2.85E-6	1	3	161	1.00E-4	1	1	113	2.38E-4
0	11	1521	2.09E-7	2	3	449	2.23E-5	2	1	353	2.19E-5
0	15	3025	3.40E-8	3	3	1153	5.82E-6	3	1	961	5.71E-6
0	19	5041	8.61E-9	4	3	2817	6.58E-7	4	1	2433	6.57E-7
0	23	7569	3.00E-9	5	3	6657	2.16E-7	5	1	5889	2.16E-7
0	27	10609	1.20E-9	6	3	15361	1.59E-8	6	1	13825	1.59E-8
0	31	14161	2.15E-10	7	3	34817	6.15E-9	7	1	31745	6.15E-9

The solution in Example 1 is analytic, it is expected that the sgSEM-N-F (full grid method) will give the best convergence rate, which is what we observe in Table 1. We also say that all methods converge exponentially fast. This validates the accuracy of the algorithms.

Example 2 is used to test the convergence behavior for solutions with corner singularity. Since the series (62) converges very slow, we use the numerical solution obtained on a fine grid as reference to calculate the errors on coarse girds. In this example, the domain $[-1, 1]^2$ is split into 16 equal squares. The numerical results are given in Table 2. For this example with a weak singularity at the corners, the full grid spectral element method still gives better convergence results but the difference now is very small. On the other hand, the sgSEM-Nm converges faster than sgSEM-N.

In Example 3, the exact solution has different singularities on $x_1 = 0$, $x_2 = 0$, and $x_1 = 0.5$, $x_2 = 0.5$. For this example, sparse grid spectral element methods deliver better convergence than the full grid spectral method. The result is showed in

Figure 4. We observe that the sgSEM-N using 4 elements give the best convergence results, with the convergence behavior of sgSEM-Nm being very close to sgSEM-N.

Example 4 is a case with a non-tensor product domain. The exact solution has singularities on $x_1 = 0$, $x_2 = 0$, and $x_1 = -0.5$, $x_2 = 0.5$. Numerical result is shown in Figure 5:(b). We observe that the sgSEM-N and sgSEM-Nm methods give very similar convergence rates, both are better than the full grid case.

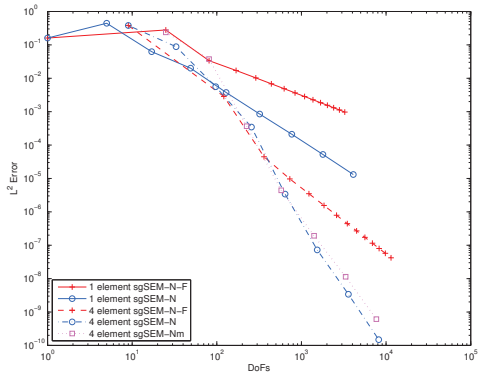
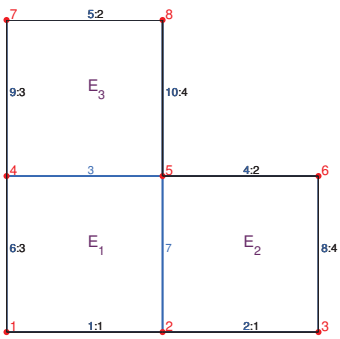
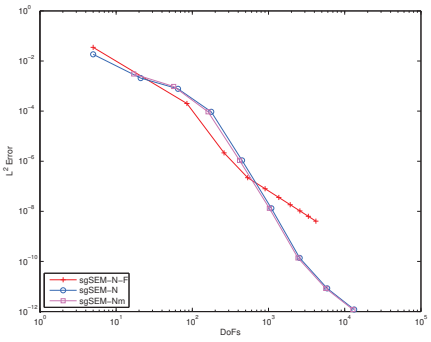


FIGURE 4. The L^2 errors of sgSEM-N-F, sgSEM-N and sgSEM $-$ Nm using different number of elements for Example 3. The number of edge nodes for sgSEM-Nm is $m + 1 = m_{l-1}$.



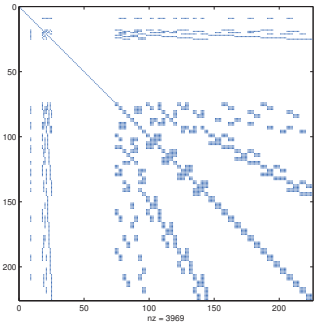
(a) A L-shape domain and a three-element partition



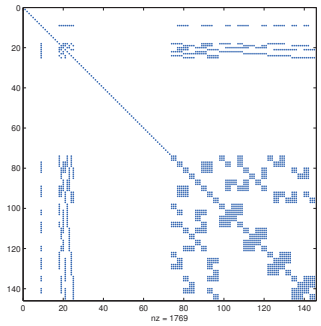
(b) Numerical results on the L-shape domain

FIGURE 5. (a) The L-shape domain and a three-element partition used in Example 4. The coordinates for vertex $1, 3, 7$ are $(-1, -1), (1, -1), (-1, 1)$, correspondingly. (b) The L^2 errors of sgSEM-N-F, sgSEM-N and sgSEM-Nm in Example 4. The number of edge nodes for sgSEM-Nm is $m + 1 = m_{l-1}$.

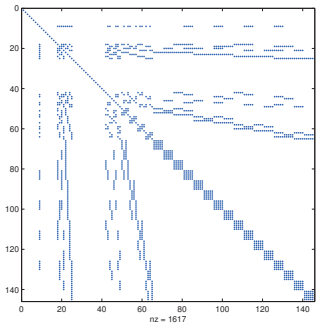
For the computational cost, since we use a spectral element approach, the system matrix and its Schur-complement have a very special sparsity patterns (See Figure 6), even though nodal local bases lead to dense local matrices. The Schur-complement can be efficiently solved by a PCG method. In Table 3, we present



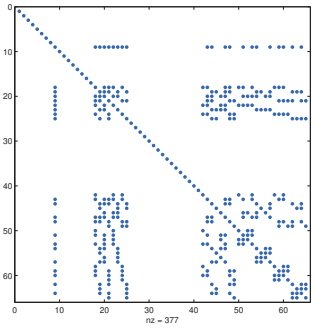
(a) System matrix of sgSEM-N



(b) Schur-complement matrix of sgSEM-N



(c) System matrix of sgSEM-Nm



(d) Schur-complement matrix of sgSEM-Nm

FIGURE 6. The sparsity patterns of discretized system matrix A for equation (26) and its Schur-complement $A_{11} - A_{12}A_{22}^{-1}A_{21}$ in sgSEM-N scheme and sgSEM-Nm scheme. Here a grid with 4×4 elements is used. Sparse grid $\mathcal{X}_3^2$ are used for all elements. Note that the boundary points are also included as DoFs.

TABLE 3. The iteration number of PCG for sgSEM-N-F, sgSEM-N and sgSEM-Nm with the block diagonal preconditioner. Schur complement is used. The column with title n_S is the sizes of Schur complement. The tolerance of stop criteria is set to 10^{-14}.

sgSEM-N-F($l = 0$)					sgSEM-N($m_0 = 3$)					sgSEM-Nm($m_0 = 1$)				
K	m_0	DoFs	n_S	n_{iter}	K	l	DoFs	n_S	n_{iter}	K	l	DoFs	n_S	n_{iter}
4	7	121	21	3	4	0	9	5	3	4	0			
	19	1225	69	3		2	97	29	4		2	81	13	3
	31	3481	117	3		4	641	125	6		4	577	61	4
	43	6889	165	3		6	3585	509	6		6	3329	253	6
	55	11449	213	3		8	18433	2045	7		8	17409	1021	6
16	3	225	33	34	16	0	225	33	34	16	0			
	7	2209	129	28		1	705	81	28		1	481	33	33
	11	6241	225	28		2	1921	177	29		2	1473	81	30
	15	12321	321	28		3	4865	369	27		3	3969	177	28
	19	20449	417	28		4	11777	753	29		4	9985	369	28
	27	42849	609	26		5	27649	1521	28		5	24065	753	29

iteration numbers of using block diagonal preconditioner for the Schur-complement system (44). We observe that the numbers of iterations slowly increase as l and K increases in the sparse grid cases, which indicates that the PCG with the block diagonal preconditioner works reasonably well.

6. Conclusions

We developed sparse grid spectral element methods using nodal bases for multi-dimensional elliptic PDEs. We use Chebyshev-Gauss-Lobatto sparse grid points to interpolate data, and use Lagrange bases in sparse grid approximation space to form linear algebraic system using a pseudo-spectral approach. The two sparse grid methods, sgSEM-N and sgSEM-Nm distinguish from each other on how many edge DoFs used. sgSEM-Nm uses less edge DoFs than sgSEM-N to get a better conditioned Schur system. Preliminary numerical methods show that the iteration numbers of sgSEM-N and sgSEM-Nm using CG with a simple block-diagonal preconditioner for the corresponding Schur-complement are similar. How to build more effective preconditioners for sgSEM-N and sgSEM-Nm will be addressed in a future work.

Since the main cost in a spectral element method is usually associated with solving the Schur-complement system, and sgSEM-Nm uses less edge DoFs, particularly in higher-dimensions and with larger numbers of elements, it can be expected that sgSEM-Nm, with a better preconditioner, can potentially be much more effective than sgSEM-N and sgSEM-N-F.

Even though we only implemented the algorithms for two dimensional case, the proposed methods work for high-dimensional cases. We will use them to study practical higher-dimensional equations in future works.

Acknowledgments

The work of Z.J. Rong was partially supported by NSFC Grant 11201393 and Fujian Provincial Natural Science grant 2013J05019. The work of J. Shen was partially supported by AFOSR grant FA9550-16-1-0102, and by NSF grant DMS-1620262. The work of H. Yu was partially supported by China National Program on Key Basic Research Project 2015CB856003 and NNSFC grant 91530322, 11371358, 11101413.

References

[1] B. A. Alpert and V. Rohklin. A fast algorithm for the evaluation of Legendre expansions. SIAM J. Sci. Comput., 12:158179, 1991.

[2] Robert Balder and Christoph Zenger. The solution of multidimensional real Helmholtz equations on sparse grids. SIAM J. Sci. Comput., 17(3):631–646, May 1996.

[3] V. Barthelmann, E. Novak, and K. Ritter. High dimensional polynomial interpolation on sparse grids. Advances in Computational Mathematics, 12(4):273288, 2000.

[4] Fischer Black and Myron Scholes. The pricing of options and corporate liabilities. The Journal of Political Economy, pages 637–654, 1973.

[5] John P. Boyd. Multipole expansions and pseudospectral cardinal functions: A new generalization of the fast Fourier transform. Journal of Computational Physics, 103(1):184–186, November 1992.

[6] H. J. Bungartz. An adaptive poisson solver using hierarchical bases and sparse grids. In Iterative Methods in Linear Algebra, pages 293–310. Amsterdam: North-Holland, 1992.

[7] Hans-Joachim Bungartz and Michael Griebel. Sparse grids. Acta Numerica, 13:1–123, 2004.

[8] C. Canuto, M. Y. Hussaini, A. Quarteroni, and T. A. Zang. Spectral methods. Scientific Computation. Springer, Berlin, 2007. Evolution to complex geometries and applications to fluid dynamics.
[9] W.S. Don and D. Gottlieb. The Chebyshev-Legendre method: implementing Legendre methods on Chebyshev points. SIAM J. Numer. Anal., 31:1519–1534, 1994.
[10] Thomas Gerstner and Michael Griebel. Numerical integration using sparse grids. Numerical Algorithms, 18(3):209–232, January 1998.
[11] M. Griebel and J. Hamaekers. Sparse grids for the schrödinger equation. Mathematical Modelling and Numerical Analysis, 41(2):215247, 2007.
[12] Ben-Yu Guo and Hong-Li Jia. Spectral method on quadrilaterals. Math. Comp., 79 (272) 2237–2264, 2010.
[13] K. Hallatschek. Fouriertransformation auf dnnen gittern mit hierarchischen basen. Numerische Mathematik, 63(1):8397, 1992.
[14] George Em Karniadakis and Spencer J. Sherwin. Spectral/hp element methods for CFD. Oxford University Press, 1999.
[15] E. Schrödinger. An undulatory theory of the mechanics of atoms and molecules. Phys. Rev., 28(6):1049–1070, December 1926.
[16] J. Shen. Efficient Chebyshev-Legendre Galerkin methods for elliptic problems. In Proceedings of ICOSAHOM, volume 95, page 233239. Citeseer, 1996.
[17] Jie Shen and L.L. Wang. Sparse spectral approximations of high-dimensional problems based on hyperbolic cross. SIAM J. Numer. Anal., 48(4):1087–1109, 2010.
[18] Jie Shen and Haijun Yu. Efficient spectral sparse grid methods and applications to high-dimensional elliptic problems. SIAM Journal on Scientific Computing, 32(6):3228–3250, 2010.
[19] Jie Shen and Haijun Yu. Efficient spectral sparse grid methods and applications to high-dimensional elliptic equations II. unbounded domains. SIAM Journal on Scientific Computing, 34(2):1141–1164, 2012.
[20] S. A Smolyak. Quadrature and interpolation formulas for tensor products of certain classes of functions. Soviet mathematics, 4:240–243, 1963.
[21] H. Yserentant. On the regularity of the electronic schrödinger equation in Hilbert spaces of mixed derivatives. Numerische Mathematik, 98(4):731759, 2004.
[22] H. Yserentant. Sparse grid spaces for the numerical solution of the electronic schrödinger equation. Numerische Mathematik, 101(2):381389, 2005.

Fujian Provincial Key Laboratory on Mathematical Modeling & High Performance Scientific Computing and School of Mathematical Sciences, Xiamen University, Xiamen, Fujian 361005, P. R. China
E-mail: zjrong@xmu.edu.cn

Department of Mathematics, Purdue University, West Lafayette, IN 47906-1957
E-mail: shen7@purdue.edu

NCMIS & LSEC, Institute of Computational Mathematics and Scientific/Engineering Computing, Academy of Mathematics and Systems Science, Beijing 100190, China

School of Mathematical Sciences, University of Chinese Academy of Sciences, Beijing 100049, China
E-mail: hyu@lsec.cc.ac.cn