# Least-Squares Finite Element Methods via Randomised Linear Algebra

Chenghao Dong

Kellogg College

University of Oxford

A thesis submitted for the degree of

*Master of Science in*
*Mathematical Modelling and Scientific Computing*

Trinity 2024

# Acknowledgements

Before starting this dissertation, I want to express my sincere gratitude to my two great supervisors, Prof. Yuji Nakatsukasa and Dr. Carolina Urzúa-Torres, for their constant support and encouragement. I would like to thank Yuji for the opportunity he offered me to work on this project and for all his inspiring ideas along the way. I am also likewise thankful to Carolina, who provided me with a lot of useful suggestions on finite elements. This project would not have progressed so smoothly without their motivation and professional advice.

I would also like to acknowledge Dr. Kathryn Gillow, our course director and my academic supervisor, for her effort to make my MMSC journey such a fruitful and memorable experience. I will always remember the moment when I first received my MMSC admission from her last year, and being able to participate in this programme would definitely be a critical stage for me to pursue my future academic career.

My heartfelt thanks also belong to all my friends from XJTLU and here at Oxford. They are one of the most empathetic, respectful, and outstanding groups of peers I have ever met in my life. My college time could have been far less colourful and impressive without their support and company.

Finally, I can never put into words my gratitude for the support that my parents and grandparents have given me throughout my life journey, whether in terms of daily life, encouragement, or financial assistance. Without them, being able to study at Oxford would only be an unattainable dream for a kid from a small county in China, and they are the ones who eventually made this dream come true.

# Abstract

The least-squares finite element methods (LSFEMs) are proposed as a modification for the traditional finite element methods in solving elliptic boundary value problems to avoid dealing with intricate stability conditions, whose solution in the discredited trial spaces can be obtained via least-squares finite element collocation methods (LSFECMs) without the necessity to explicitly evaluate the integrals.

It is known that LSFECM can achieve quasi-optimal convergence under sophisticatedly picked collocation points, making it more of an art than a universally applicable method. One way to address this barrier is provided via oversampling, which usually leads to extremely over-determined algebraic systems. In this dissertation, we focused on whether the techniques from randomised linear algebra, especially sketch-and-solve, can be utilised to accelerate the solving process of these over-determined systems. We also investigate how oversampling could ensure the quasi-optimal convergence of LSFECMs and discuss the possible impact that sketch-and-solve may have on the overall convergence rate.

The thesis is structured as follows: In the first chapter, we will briefly review the related work in the past literature, describe our main contributions and specify some notations which will be used in our later discussion. Then, we will turn to the theoretical foundations upon which the analyses in the following chapters are built, in Chapters 2 & 3 for LSFECM and sketch-and-solve. Chapter 4 will provide a numerical result for the application of sketch-and-solve to LSFECMs via two respective examples in 1D and 2D. Next, a detailed convergence and error analysis will be presented with numerical evidence in Chapter 5, constituting the main theoretical contributions of this dissertation. Finally, the conclusion and possible directions for future research will be given in Chapter 6.

# Contents

# Chapter 1

# Introduction

## 1.1 Literature Overview

Numerical methods for solving linear elliptic boundary value problems, such as Poisson equations and Navier-Stokes equations, have long been a popular topic across various fields, among which the method of finite element (FEM) has been widely considered due to its solid mathematical foundation in variational principles, and its advantages in dealing with geometrically complex domains and in relaxing regularity requirements.

In traditional finite element methods, boundary value problems are usually cast into a set of weak equations using integration by parts. In some instances, these weak equations are equivalent to the minimization of certain quadratic energy functionals where variational principles lead to symmetric and strongly coercive bilinear forms [19, §6]. The well-posedness of these weak problems and their conforming discretisations is then guaranteed by the Lax–Milgram theorem [19, §4]. In most cases, especially for elliptic boundary value problems involving partial differential equations with multiple variables, classical formulations often fall into saddle-point optimizations [5]. The discretisation for these weak problems should then satisfy a more complex inf-sup or Ladyzhenskaya–Babuska-Brezzi (LBB) condition for well-posedness. This usually implies finite element spaces for dependent variables cannot be chosen independently; see [5] and [19, §13]. A well-studied example is given by the Stokes equation [19, §14]. Moreover, classical formulations in these situations usually result in indefinite discrete algebraic problems, e.g., [6, §1.3].

The least-squares finite element method (LSFEM) was then proposed as an attempt to circumvent these intricate stability conditions by introducing the least-squares principle into FEMs, leading to the minimization of some quadratic, norm-equivalent functional, whose corresponding variational problems and discretisations are both well-posed. There are two approaches to obtaining the solution of LSFEM on the discretised solution space: the *continuous formulation*, which is sometimes referred to as the *bona fide least-squares method* [6, §1], and the *discrete formulation*, often named as the (*over-determined*) *collocation* [18], or the *least-squares finite element collocation method* (LSFECM) [25, §4.11] if the discretised trial space is a finite element space. In the later formulation, the discretisation is usually taken before applying the least squares principle.

Unlike conventional continuous formulations, the collocation method does not require explicit evaluations for the integrals while assembling the algebraic system, which results in a significant computational advantage. However, the methods typically lead to over-determined algebraic systems, where the degree of over-determination largely depends on the number of collocation points selected. Usually, practitioners may aim to select a proper type and number of collocation points to ensure that the final algebraic system is almost square and that the approximate solution can converge quasi-optimally with respect to the mesh size as the one obtained via the continuous formulation [10, §3] [25, §4.11]. Hence, choosing the collocation points usually turns out to be more of an art than a universally applicable method for all scenarios.

Therefore, to further improve the usability of the collocation method, one natural idea is to introduce sketching, i.e., subspace embeddings (see [31, §8], [34, §2] and [41, §2]), while oversampling collocation points to reduce the final dimensionality of the derived algebraic system. On the one hand, oversampling collocation points may ensure the non-singularity of the normal equation for the final over-determined system while also improving the accuracy of the method. As we will show later, the essence of collocation is to approximate the integrals in the continuous formulation with point-wise evaluations. On the other hand, bringing in sketching can also speed up the application of the oversampling collocation and help balance the number of equations with that of the unknowns. To the best of our knowledge, no one has yet attempted this “oversampling LSFECM + sketching” framework in past literature.

## 1.2 Main Contributions

The main contributions of this dissertation are as follows: **(1)** We build from scratch a MATLAB toolkit for the "oversampling LSFECM + sketching" framework to solve 1D/2D first-order linear boundary value problems (Chapter 2 & 3) and evaluate the effect of 12 sketches on solving linear least-squares problems derived from oversampling LSFECMs in both 1D and 2D scenarios (Chapter 4). **(2)** We proved the sufficient condition for oversampling LSFECMs to achieve quasi-optimal convergence when applied to fully $H^1$-coercive first-order linear boundary value problems with zero Dirichlet boundary conditions (Section 5.2) and provide a rule of thumb to estimate method parameters in a more general case (Section 5.3 & 5.4). **(3)** We test the convergence of LSFECM on two example problems and discuss the possible impact that sketching may have on the overall convergence rate (Section 5.5 & 5.6).

In addition to the above main contributions, this dissertation also improves the method of fast leverage score approximation proposed by Drineas et al. in [16, §3], where we replace the subsampled randomized Hadamard transform in the original algorithm with Hashing embeddings (Section 3.2.2). In Chapter 4, we experimentally demonstrate a significant speedup in the runtime for the new algorithm to produce approximate leverage scores with similar accuracy, i.e., misestimation factors. Finally, we compare the method of whole sketching with block sketching in Chapter 4 and further discuss the advantages of the latter approach in supporting parallel computation and maintaining structures for the linear algebraic systems derived in the context of LSFECMs.

## 1.3 General Notations

### 1.3.1 Functional Analysis

Throughout this dissertation, bold uppercase and lowercase letters, such as $\mathbf{A}$ and $\mathbf{a}$, are used to represent matrices and column vectors, respectively, unless otherwise specified. For an arbitrary vector space $X$, we write $(\cdot,\cdot)_X$ and $\|\cdot\|_X$ to denote its inner product and norm. The product space of two spaces $X$ and $Y$ is offten written as $X \times Y$ with an inner product given by $(\cdot,\cdot)_{X\times Y} = (\cdot,\cdot)_X + (\cdot,\cdot)_Y$. Based on these notations, we define $X^k = \underbrace{X \times \cdots \times X}_{k \text{ times}}$.

For some (Lipschitz) domain $\Omega$ with a boundary $\Gamma := \partial\Omega$, $C^k(\Omega)$ is used for the space of functions defined over $\Omega$ which are $k$ times continuously differentiable, and $L^2(\Omega)$ is for space of functions which are square integrable. We also follow the standard notation and definition for the Sobolev spaces $H^k(\Omega)$ of functions whose (weak) derivatives up to order $k$ all belong to $L^2(\Omega)$, with corresponding inner products $(\cdot,\cdot)_{k,\Omega}$ and norms $\|\cdot\|_{k,\Omega}$. For $k = 0$, we assign $H^0(\Omega) := L^2(\Omega)$.

For vectorised functions, we would write $\mathbf{L}^2(\Omega)$, $\mathbf{H}^k(\Omega)$ instead of $[L^2(\Omega)]^n$, $[H^k(\Omega)]^n$, and keep denoting the corresponding inner products and norms by $(\cdot,\cdot)_{k,\Omega}$ and $\|\cdot\|_{k,\Omega}$, respectively, when there is no risk of confusion. The trace space $\mathbf{H}^{1/2}(\Gamma)$ is defined as the space consisting of traces of functions belonging to $\mathbf{H}^1(\Omega)$. The norm for $\mathbf{g} \in \mathbf{H}^{1/2}(\Gamma)$ is

$$\|\mathbf{g}\|_{1/2,\Gamma} = \inf_{\mathbf{u}\in\mathbf{H}^1(\Omega)\ ,\ \mathbf{u}|_\Gamma=\mathbf{g}} \|\mathbf{u}\|_{1,\Omega}; \tag{1.3.1}$$

see [25, §4.1]. We may also encounter the following spaces as in [6, §1],

$$\mathbf{H}(\Omega, \mathrm{div}) = \{\ \mathbf{v} \in \mathbf{L}^2(\Omega)\ :\ \nabla\cdot\mathbf{v} \in L^2(\Omega)\ \}, \tag{1.3.2}$$

$$\mathbf{H}(\Omega, \mathrm{curl}) = \{\ \mathbf{v} \in \mathbf{L}^2(\Omega)\ :\ \nabla\times\mathbf{v} \in \mathbf{L}^2(\Omega)\ \}, \tag{1.3.3}$$

and

$$\mathbf{H}^1_0(\Omega) = \{\ \boldsymbol{\phi} \in \mathbf{H}^1(\Omega)\ :\ \boldsymbol{\phi}|_{\partial\Omega} = \mathbf{0}\ \}, \tag{1.3.4}$$

$$\mathbf{H}_0(\Omega, \mathrm{curl}) = \{\ \mathbf{v} \in \mathbf{H}(\Omega, \mathrm{curl})\ :\ \mathbf{v}\times\mathbf{n}|_{\partial\Omega} = 0\ \}. \tag{1.3.5}$$

### 1.3.2 Linear Algebra

When there is no risk of ambiguity, the $p$-norm of a vector and the corresponding induced $p$-norm of a matrix are both written as $\|\cdot\|_p$. Additionally, the Frobenius norm of a matrix is written as $\|\cdot\|_F$. For a given $m\times n$ real matrix $\mathbf{A} \in \mathbb{R}^{m\times n}$, we use $\mathbf{A}^{\mathrm{T}}$ to denote its transpose, $\mathbf{A}^\dagger$ to indicate its pseudo-inverse, and $\kappa_p(\mathbf{A})$ to represent its condition number with respect to the induced p-norm $\|\cdot\|_p$. For the assembly of submatrices and vectors, we use ";" to denote stacking by rows and "," to denote stacking by columns. For instance, given a set of matrices $\{\mathbf{A}_i\}_{i=1}^n$, we have

$$[\mathbf{A}_1\ ;\ \cdots\ ;\ \mathbf{A}_n] := \begin{bmatrix}\mathbf{A}_1\\ \vdots\\ \mathbf{A}_n\end{bmatrix} \quad \text{and} \quad [\mathbf{A}_1\ ,\ \cdots\ ,\ \mathbf{A}_n] := \begin{bmatrix}\mathbf{A}_1 & \cdots & \mathbf{A}_n\end{bmatrix}.$$

Specifically, we use $\mathrm{vec}(\mathbf{A})$ as a shorthand notation for $[\mathbf{a}_1\ ;\ \cdots\ ;\ \mathbf{a}_n]$, where $\mathbf{a}_j$ are column vectors of the matrix $\mathbf{A}$. We may also adhere to MATLAB's standard syntax to perform element extraction from matrices and vectors. For example, given a set of row and column indices $\mathbf{i} = [i_1, \cdots, i_p]$ and $\mathbf{j} = [j_1, \cdots, j_q]$, we define

$$\mathbf{A}(\mathbf{i}, \mathbf{j}) = \begin{bmatrix} a_{i_1,j_1} & \cdots & a_{i_1,j_q} \\ \vdots & \ddots & \vdots \\ a_{i_p,j_1} & \cdots & a_{i_p,j_q} \end{bmatrix},$$

where $a_{i,j}$ ($\mathbf{A}_{i,j}$) is used for the entry of $\mathbf{A}$ at the $ith$ row and the $jth$ column. For some $a, b \in \mathbb{N}_+$, $a < b$, we use $a : b$ to represent the row vector $[a, a+1, \cdots, b]$.

We also outline the following two decompositions, which will be used in our later discussion.

**Definition 1.3.1. QR Decomposition** [7]
*For a given matrix* $\mathbf{A} \in \mathbb{R}^{m\times n}$, *the* QR *decomposition is given by*

$$\mathbf{A} = \mathbf{QR}, \tag{1.3.6}$$

*where* $\mathbf{Q} \in \mathbb{R}^{m\times n}$ *has orthonormal columns and* $\mathbf{R} \in \mathbb{R}^{n\times n}$ *is upper triangular. The complexity of applying regular* QR *decomposition is* $\mathcal{O}(mn^2)$.

**Definition 1.3.2. Compact SVD** [15, §7.8]
*The compact singular value decomposition* (SVD) *for a given matrix* $\mathbf{A} \in \mathbb{R}^{m\times n}$ *with* $\text{rank}(\mathbf{A}) = r$ *is*

$$\mathbf{A} = \mathbf{U}_r\mathbf{\Sigma}_r\mathbf{V}_r^{\text{T}}, \tag{1.3.7}$$

*where* $\mathbf{\Sigma}_r \in \mathbb{R}^{r\times r}$ *is a full rank diagonal matrix and* $\mathbf{U}_r \in \mathbb{R}^{m\times r}$, $\mathbf{V}_r \in \mathbb{R}^{n\times r}$ *both has orthonormal columns. The compact* SVD *of a matrix with* $m \geq n$ *can be obtained via an economy-size* SVD *through* $\mathcal{O}(mn^2)$ *flops; see* [38].

### 1.3.3 Other Notations

For some $n \in \mathbb{N}_+$, we define $[n] := \{1, 2, \cdots, n\}$. The notations $\Omega(\cdot)$, $\mathcal{O}(\cdot)$ and $\Theta(\cdot)$ are used to indicate the lower bound, the upper bound, and both bounds up to a certain order, respectively. The relation $f \leq C \cdot g$ for some positive constant $C > 0$ is often written as $f \lesssim g$. Other notations used in this thesis will be explained once they are first proposed; also see Appendix A.1 for a list of important notations used throughout this dissertation.

# Chapter 2

# Least-Squares Finite Element Collocation Methods

In this chapter, we will introduce the general formulation for the Least-Squares Finite Element Collocation Method (LSFECM), discuss its application to solving general linear boundary-value problems, and concentrate on constructing the linear least-squares problem central to this thesis for both 1D and 2D implementations.

## 2.1 General Formulation

### 2.1.1 Least-Squares Principles

For convenience, we restrict ourselves to solving the following linear boundary value problem: let $\mathcal{A}$ and $\mathcal{B}$ be two linear differential operators and consider finding a solution $\mathbf{u}$ such that it satisfies

$$\mathcal{A}[\mathbf{u}(\mathbf{x})] = \mathbf{f}(\mathbf{x}) \quad \text{for} \quad \mathbf{x} \in \Omega \subset \mathbb{R}^{n_d}, \tag{2.1.1a}$$

$$\mathcal{B}[\mathbf{u}(\mathbf{x})] = \mathbf{g}(\mathbf{x}) \quad \text{for} \quad \mathbf{x} \in \Gamma := \partial\Omega, \tag{2.1.1b}$$

over some bounded Lipschitz domain $\Omega$. As we have already discussed in the review of Chapter 1, solving the above problem with traditional finite element methods may eventually encounter complex stability conditions, which can be avoided by turning to finite element methods of least-squares type (LSFEM), where a quadratic norm-equivalent functional is minimised to ensure the well-posedness of the formulation. To proceed, we define the true solution to (2.1.1) in terms of the LSFEM as follows.

**Definition 2.1.1. Solution of the LSFEM** [6, §2]
*Let $\Omega$ be a bounded Lipschitz domain, $X(\Omega)$, $Y(\Omega)$ and $Y(\Gamma)$ be Hilbert spaces with corresponding norms $\|\cdot\|_{X(\Omega)}$, $\|\cdot\|_{Y(\Omega)}$ and $\|\cdot\|_{Y(\Gamma)}$, respectively, such that $(\mathbf{f}, \mathbf{g}) \in Y(\Omega) \times Y(\Gamma)$ and $\|\mathbf{f}\|_{Y(\Omega)}$, $\|\mathbf{g}\|_{Y(\Gamma)}$ are bounded from above. Further, assume that for all $\mathbf{u} \in X(\Omega)$, the mapping $(\mathcal{A}[\cdot], \mathcal{B}[\cdot]) : X(\Omega) \mapsto Y(\Omega) \times Y(\Gamma)$, forms a homeomorphism, thus satisfying the energy balance relation, i.e.,*

$$\|\mathbf{u}\|_{X(\Omega)} \ \lesssim \ \|\mathcal{A}[\mathbf{u}]\|_{Y(\Omega)} + \|\mathcal{B}[\mathbf{u}]\|_{Y(\Gamma)} \ \lesssim \ \|\mathbf{u}\|_{X(\Omega)} \ . \tag{2.1.2}$$

*Then, $\mathbf{u} \in X(\Omega)$ is a "true" solution to* (2.1.1) *under* LSFEM *if it minimises the norm-equivalent quadratic least-squares functional*

$$\mathcal{I}[\mathbf{u}] = \frac{1}{2}( \ \|\mathcal{A}[\mathbf{u}] - \mathbf{f}\|^2_{Y(\Omega)} + \|\mathcal{B}[\mathbf{u}] - \mathbf{g}\|^2_{Y(\Gamma)} \ ). \tag{2.1.3}$$

From now on, our discussion throughout the thesis will always be based on the assumptions made in the above definition. When $X(\Omega)$ and $Y(\Omega) \times Y(\Gamma)$ can be chosen as products of $\mathbf{H}^1$- and $\mathbf{L}^2$-spaces, respectively, to satisfy the energy balance relation (2.1.2), the formulation is said to be *"fully" $H^1$-coercive*; see [6, §2]. The relation (2.1.2) is crucial to the well-posedness of the minimization problem (2.1.3), as given in the Theorem 2.1.1 below. For proof, see Theorem A.2.1 in Appendix A.2.

**Theorem 2.1.1. Well-posedness of the LSFEM** [6, §2]
*Let $\mathcal{A}[\cdot]$, $\mathcal{B}[\cdot]$, $\mathbf{f}$, $\mathbf{g}$, $\Omega$, $X(\Omega)$, $Y(\Omega)$, $Y(\Gamma)$ and $\mathcal{I}[\mathbf{u}]$ be as defined in Definition 2.1.1. Then, minimizing $\mathcal{I}[\mathbf{u}]$ is equivalent to finding $\mathbf{u} \in X(\Omega)$ such that*

$$\mathrm{b}(\mathbf{u}, \mathbf{v}) = F(\mathbf{v}) \quad \text{for all} \quad \mathbf{v} \in X(\Omega), \tag{2.1.4}$$

*where*

$$\mathrm{b}(\mathbf{u}, \mathbf{v}) := (\mathcal{A}[\mathbf{u}], \mathcal{A}[\mathbf{v}])_{Y(\Omega)} + (\mathcal{B}[\mathbf{u}], \mathcal{B}[\mathbf{v}])_{Y(\Gamma)}, \tag{2.1.5}$$

$$F(\mathbf{v}) := (\mathbf{f}, \mathcal{A}[\mathbf{v}])_{Y(\Omega)} + (\mathbf{g}, \mathcal{B}[\mathbf{v}])_{Y(\Gamma)}. \tag{2.1.6}$$

*Furthermore, the bilinear form $\mathrm{b}(\cdot, \cdot)$ is both continuous and $X(\Omega)$-coercive; thus the problem* (2.1.4) – (2.1.6) *is well-posed by the Lax-Milgram theorem.*

Let $X_h(\Omega)$ be a (finite-dimensional) subspace of $X(\Omega)$ where an approximated solution $\mathbf{u}_h(\mathbf{x}; \mathbf{c})$ will be sought instead of $\mathbf{u}(\mathbf{x})$ with parameters contained in vector $\mathbf{c}$, and define the interior and boundary residuals $R_\Omega(\mathbf{x}; \mathbf{c})$ and $R_\Gamma(\mathbf{x}; \mathbf{c})$ to be

$$R_\Omega(\mathbf{x}; \mathbf{c}) = \mathcal{A}[\mathbf{u}_h] - \mathbf{f} \quad \text{for} \quad \mathbf{x} \in \Omega, \tag{2.1.7}$$

$$R_\Gamma(\mathbf{x}; \mathbf{c}) = \mathcal{B}[\mathbf{u}_h] - \mathbf{g} \quad \text{for} \quad \mathbf{x} \in \Gamma. \tag{2.1.8}$$

Two formulations are presented to obtain $\mathbf{u}_h$ by further minimising the residual error. In *continuous formulations*, the residual error is defined as

$$\mathcal{I}_h^c(\mathbf{c}) = \|R_\Omega(\mathbf{x};\mathbf{c})\|_{Y(\Omega)}^2 + \|R_\Gamma(\mathbf{x};\mathbf{c})\|_{Y(\Gamma)}^2, \tag{2.1.9}$$

while in *discrete formulations*, it is obtained by summing weighted squared residuals evaluated at a finite set of points $\{\mathbf{z}_i\}_{i=1}^{N_\Omega}$ and $\{\hat{\mathbf{z}}_j\}_{j=1}^{N_\Gamma}$ chosen in $\Omega$ and on $\Gamma$ correspondingly,

$$\mathcal{I}_h^d(\mathbf{c}) = \textstyle\sum_{i=1}^{N_\Omega}\|R_\Omega(\mathbf{z}_i;\mathbf{c})\|_2^2 + \sum_{j=1}^{N_\Gamma}\|\mathbf{\Lambda} R_\Gamma(\hat{\mathbf{z}}_j;\mathbf{c})\|_2^2. \tag{2.1.10}$$

Here, $\mathbf{\Lambda}$ is a diagonal matrix specifying the relative weights for each boundary residual with respect to the interior ones. The latter discrete formulation is often referred to as the *over-determined collocation* [18]. The points $\{\mathbf{z}_i\}_{i=1}^{N_\Omega}$ and $\{\hat{\mathbf{z}}_j\}_{j=1}^{N_\Gamma}$ are then called the *collocation points*; see [10, §1] and [25, §4.11]. The discrete version (2.1.10) will also be the core formulation used throughout this dissertation.

### 2.1.2 Collocation with Finite Elements

Although the form of the trial solution $\mathbf{u}_h$ may be expanded by basis functions with either global or local supports, in this thesis, we will by default focus on expanding $\mathbf{u}_h$ with functions in continuous finite element spaces $X_h^c(\Omega)$, which is known as the *least-squares finite element collocation methods* [25, §4.11], or LSFECM for short. The first step of our implementation is, therefore, to choose the proper way of discretisation for the problem domain $\Omega$. We start our discussion here by briefly recalling the definition of finite elements and continuous finite element spaces as follows.

**Definition 2.1.2. Finite Element** [19, §8]
*A finite element is a triple $(K, \mathcal{V}_K, \mathcal{L}_K)$ where $K \subset \mathbb{R}^{n_d}$ is a nonempty cell with connected interior and piecewise Lipschitz-continuous boundary; $\mathcal{V}_K$ is a finite-dimensional function space with $\dim(\mathcal{V}_K) = d_K$ defined on $K$, and the degrees of freedom $\mathcal{L}_K = \{\ell_i^K\}_{i=1}^{d_K}$ is a set of bases for the dual $\mathcal{V}_K^*$. The bases of $\mathcal{V}_K$, $\{\phi_j^K\}_{j=1}^{d_K}$, uniquely defined by setting $\ell_i^K(\phi_j^K) = \delta_{ij}$ (the Kronecker delta), are called nodal bases.*

The first step for most methods involving finite elements is usually to decompose the target (polytopic) domain $\Omega$ into a mesh $\mathcal{M}_h = \{K_i\}$, such that the closure $\overline{\Omega} = \cup_i K_i$, and the none-empty intersections $K_i \cap K_j$ for $i \neq j$ is either a common vertex or a common facet of these cells. We use $h$ to denote the size of the discretised mesh, defined as,

$$h = \max_{K\in\mathcal{M}_h} \sup\{\|\mathbf{x}-\mathbf{y}\|_2 : \mathbf{x},\mathbf{y} \in K\}. \tag{2.1.11}$$

After meshing $\Omega$, a finite element $(K, \mathcal{V}_K, \mathcal{L}_K)$ is assigned to each cell $K \in \mathcal{M}_h$, together with a corresponding local-to-global map for future assembly, defined as an injection $\iota_K : [d_K] \mapsto [N]$ to ensure

$$\ell_i^K[v|_K] = \ell_{\iota_K(i)}[v] \tag{2.1.12}$$

for $N$ global degrees of freedom $\mathcal{L}_{\text{glob}} = \{\ell_i\}_{i=1}^N$. We are now ready to conclude the definition of continuous finite element spaces.

**Definition 2.1.3. Continuous Finite Element Spaces**
*A continuous finite element space generated by the mesh $\mathcal{M}_h$ for the target domain $\Omega$ is a function space $X_h^c(\Omega)$ where local-to-global mappings $\{\iota_K\}_{K\in\mathcal{M}}$ map the matching local degrees of freedom to the same global degrees of freedom*

$$X_h^c(\Omega) = \left\{ v \in X_h^{\text{brk}}(\Omega) \ : \ \ell_i^K[v|_K] = \ell_{i'}^{K'}[v|_{K'}] \iff \iota_K(i) = \iota_{K'}(i') \right\}, \tag{2.1.13}$$

*where $X_h^{\text{brk}}(\Omega) = \{v : v|_K \in \mathcal{V}_K,\ \forall K \in \mathcal{M}_h\}$ is the broken finite element space.*

Only linear Lagrange elements will be used to solve all the problems proposed in this thesis. The definition for such elements is given below.

**Definition 2.1.4. Lagrange Elements [19, §8]**
*A Lagrange element $\text{CG}_q$ of degree $q$ is a finite element $(K, \mathcal{V}_K, \mathcal{L}_K)$ where $K$ is a simplex, $\mathcal{V}_K = \mathcal{P}_q(K)$ and $\mathcal{L}_K = \{\ell_i^K\}_{i=1}^{[n_q]}$ such that*

$$\ell_i^K[v] = v(\mathbf{x}_i). \tag{2.1.14}$$

*Here, $\mathcal{P}_q(K)$ is the space of $q$-degree polynomials defined on $K$, $n_q = \dim(\mathcal{P}_q(K))$ and $\{\mathbf{x}_i\}_{i=1}^{n_q}$ is an enumeration of points in the element.*

Although following the least-square principles allows us to avoid dealing with complex stability conditions in the traditional FEM while determining the subspace $X_h(\Omega)$, one visible disadvantage of applying such a formulation is that it does not naturally reduce the regularity requirements for the solution space $X(\Omega)$ (and the trial space $X_h(\Omega)$) as the traditional FEM does. Therefore, one essential requirement for building practical least-squares formulations which prevail over their traditional counterparts is to use standard and easy-to-use finite element spaces [6, §2]. We should thus restrict ourselves to using only linear Lagrange elements $\text{CG}_1$. The 2D discretisation using $\text{CG}_1$ could be quickly implemented with the function **generateMesh** in MATLAB's Partial Differential Equation Toolbox [22].

Clearly, applying these linear elements will only form finite element spaces that are $C^0$-conforming. Although this may be enough for solving first-order problems over bounded domains (see Theorem 5.2 in [8, §5]), where the solutions are only required to be in $\mathbf{H}^1(\Omega)$, a decomposition technique in the next section will be needed to solve problems of higher orders. For collocation with elements of higher regularity, we refer readers to [10], where Hermite elements are used to form $C^1$-conforming spaces.

## 2.2 Decomposition into First-order Systems

As discussed in the previous Section 2.1.2, continuous finite element spaces $X_h^c(\Omega)$ generated by the linear Lagrange elements $\mathrm{CG}_1$ are only $C^0$-conforming, which is typically impractical to solve problems involving higher-order differential operators through direct least-square formulations. To achieve both goals of avoiding complex stability conditions in traditional methods and discretising with low-regularity finite element spaces, we have to decompose the original problems "strongly" into first-order systems. This is usually achieved by introducing extra dependent variables. We will illustrate the core idea of this decomposition method with an example, which will also be used later in the numerical experiment. Consider the 2D Poisson equation

$$-\triangle\varphi = f \quad \text{for} \quad (x, y) \in \Omega, \tag{2.2.1a}$$

$$\varphi = g \quad \text{for} \quad (x, y) \in \Gamma. \tag{2.2.1b}$$

By introducing $\mathbf{w} = \nabla\phi$, and observing that $\triangle\varphi = \nabla\cdot(\nabla\varphi) = \nabla\cdot\mathbf{w}$, we can decompose (2.2.1) into

$$\nabla\varphi - \mathbf{w} = 0 \quad \text{in} \quad \Omega, \tag{2.2.2a}$$

$$-\nabla\cdot\mathbf{w} = f \quad \text{in} \quad \Omega, \tag{2.2.2b}$$

$$\nabla\times\mathbf{w} = 0 \quad \text{in} \quad \Omega, \tag{2.2.2c}$$

$$\varphi = g \quad \text{on} \quad \Gamma, \tag{2.2.2d}$$

$$\mathbf{w}\times\mathbf{n} = \nabla g\times\mathbf{n} \quad \text{on} \quad \Gamma. \tag{2.2.2e}$$

From here and forward, we shall refer to the unit outward normal on the boundary $\Gamma$ by $\mathbf{n}$. The equation (2.2.2c) is known as the *curl constraint*, which is a key augmentation to ensure the full $H^1$-coercivity of the final system [6, §3.1]. It is also natural to add such a constraint due to the fact that the gradient fields are irrotational.

Finally, since $\varphi = g$ has been specified on $\Gamma$, $\nabla(\varphi - g)\times\mathbf{n} = 0$ and therefore the new variable should satisfy $\mathbf{w}\times\mathbf{n} = \nabla g\times\mathbf{n}$ across $\Gamma$, which leads to the new boundary condition (2.2.2e). When $g = 0$, the system (2.2.2) is fully $H^1$-coercive if we impose the boundary conditions strongly to the solution space $X(\Omega)$ and let

$$X(\Omega) = \{ \ (\varphi, \mathbf{w}) \ : \ (\varphi, \mathbf{w}) \in H_0^1(\Omega) \times (\mathbf{H}(\Omega, \mathrm{div}) \cap \mathbf{H}_0(\Omega, \mathrm{curl})) \ \}, \tag{2.2.3}$$

since $\mathbf{H}(\Omega, \mathrm{div}) \cap \mathbf{H}_0(\Omega, \mathrm{curl})$ is both topologically and algebraically equivalent to $\mathbf{H}^1(\Omega)$ for bounded Lipschitz domain $\Omega$; see [6, §3.1] and [20, §2.3]. When $g \neq 0$ but $\Omega$ is simply connected, the system can be made equivalent to an elliptic system of Petrovsky type by adding an extra dummy variable; see [26] and [27].

## 2.3 Solving First-order Linear Systems

### 2.3.1 A General Framework

Our discussion of LSFECM for first-order linear systems begins by providing a general framework within which the implementation was taken. We will be discussing systems of first-order linear differential equations in the following form

$$\mathcal{A}[\mathbf{u}] := \sum_{i=1}^{n_d} \mathbf{A}_i \frac{\partial \mathbf{u}}{\partial x_i} + \mathbf{A}_0 \mathbf{u} = \mathbf{f} \quad \text{in} \quad \Omega, \tag{2.3.1a}$$

$$\mathcal{B}[\mathbf{u}] := \sum_{i=1}^{n_d} \mathbf{B}_i \frac{\partial \mathbf{u}}{\partial x_i} + \mathbf{B}_0 \mathbf{u} = \mathbf{g} \quad \text{on} \quad \Gamma, \tag{2.3.1b}$$

with $n_d \in \{1, 2\}$ indicating the number of dependent variables, $\Omega \subset \mathbb{R}^{n_d}$ being a Lipschitz domain, and $\Gamma = \partial\Omega$ being the boundary. $\mathbf{A}_i$, $\mathbf{B}_i$ for $i \in \{0 : n_d\}$ are matrices of functions depending only on $\mathbf{x}$ with a respective dimension of $m_\Omega \times n_u$ and $m_\Gamma \times n_u$, and $\mathbf{u}$ is a vector field having $n_u$ components.

An important subclass of these problems is the elliptic boundary value problems of the Petrovsky type. Namely, $\mathbf{A}_i \in \mathbb{R}^{2\eta \times 2\eta}$ are square matrices satisfying $\det(\sum_{i=1}^{n_d} \alpha_i \mathbf{A}_i) \neq 0$ for all non-zero real $\{\alpha_i\}_{i=1}^{n_d}$; see [6, §2] and [25, §4.7]. If $\mathcal{B}[\mathbf{u}] = \mathbf{B}_0 \mathbf{u}$, $\mathbf{B}_0 \in \mathbb{R}^{\eta \times 2\eta}$ is full rank, and satisfies the Shapiro-Lopatinskij condition [28], an a prior estimate could be obtained for problem (2.3.1); see [6, §4.2],

$$\|\mathbf{u}\|_{1,\Omega} \lesssim \|\textstyle\sum_{i=1}^{n} \mathbf{A}_i \mathbf{u}_{x_i} + \mathbf{A}_0 \mathbf{u}\|_{0,\Omega} + \|\mathbf{B}_0 \mathbf{u}\|_{1/2,\Gamma}. \tag{2.3.2}$$

The least-squares principle allows us to choose independent finite element spaces for all dependent variables. Therefore, we will set our trial space $X_h(\Omega)$ for $\mathbf{u}_h$ as $X_h(\Omega) = [X_{1,h}^c(\Omega)]^{n_u}$ where $X_{1,h}^c(\Omega)$ are identical continuous finite element spaces generated by the $\mathrm{CG}_1$ elements.

For each $X^c_{1,h}(\Omega)$, the domain will be partitioned with $n_v$ (distinct) global vertices, $\mathbb{V} = \{\mathbf{v}_i\}_{i=1}^{n_v}$, corresponding to $n_v$ global bases $\{\phi_i(\mathbf{x})\}_{i=1}^{n_v}$. The local degrees of freedom, as well as the local vertices and bases, are assembled globally across the map $\iota_K(j) : [n_d + 1] \mapsto [n_v]$ in each element $K \in \mathcal{M}_h$. We further assume the solution ansatz for $\mathbf{u}_h \in X_h(\Omega)$ to be

$$\mathbf{u}_h(\mathbf{x}) = \mathbf{C}\boldsymbol{\phi}(\mathbf{x}) = \sum_{i=1}^{n_v} \phi_i(\mathbf{x})\mathbf{c}_i, \tag{2.3.3}$$

where $\mathbf{C} = [\mathbf{c}_1, \cdots, \mathbf{c}_{n_v}] \in \mathbb{R}^{n_u \times n_v}$ represents the coefficient matrix with column vectors $\mathbf{c}_i$, and $\boldsymbol{\phi} = [\phi_1, \cdots, \phi_{n_v}]^{\mathrm{T}}$ denotes the vector of basis functions.

We could now start to evaluate the residuals. For a specific collocation point $\mathbf{z} \in K$, the value of the trial function $\mathbf{u}_h(\mathbf{z})$ is just

$$\mathbf{u}_h(\mathbf{z}) = \sum_{j=1}^{n_d+1} \phi_j^K(\mathbf{z})\mathbf{c}_j^K = \sum_{j=1}^{n_d+1} \phi_{\iota_K(j)}(\mathbf{z})\mathbf{c}_{\iota_K(j)}, \tag{2.3.4}$$

where we use $\phi_j^K$ and $\mathbf{c}_j^K \in \mathbb{R}^{n_u \times 1}$ to denote the local nodal bases and their corresponding coefficients. After applying the linear operator $\mathcal{A}[\cdot]$, we obtain

$$\mathcal{A}[\mathbf{u}_h(\mathbf{z})] = \sum_{j=1}^{n_d+1} \mathcal{A}\left(\phi_j^K\right)\mathbf{c}_{\iota_K(j)} \quad \text{with} \quad \mathcal{A}(\phi(\mathbf{z})) := \sum_{i=1}^{n_d} \frac{\partial}{\partial x_i}\phi(\mathbf{z})\mathbf{A}_i + \phi\mathbf{A}_0\ . \tag{2.3.5}$$

The interior residuals can thus be calculated as

$$R_\Omega(\mathbf{z};\mathbf{c}) = \Big[\ \mathcal{A}\left(\phi_1^K(\mathbf{z})\right) \ \ \cdots \ \ \mathcal{A}\left(\phi_{n_d+1}^K(\mathbf{z})\right)\Big] \begin{bmatrix} \mathbf{c}_{\iota_K(1)} \\ \vdots \\ \mathbf{c}_{\iota_K(n_d+1)} \end{bmatrix} - \mathbf{f}(\mathbf{z}), \tag{2.3.6}$$

for $\mathbf{c} = \mathrm{vec}(\mathbf{C})$. The case is similar for the boundary residuals, where if $\mathbf{z} \in K \cap \Gamma$ we have

$$R_\Gamma(\mathbf{z};\mathbf{c}) = \Big[\ \mathcal{B}\left(\phi_1^K(\mathbf{z})\right) \ \ \cdots \ \ \mathcal{B}\left(\phi_{n_d+1}^K(\mathbf{z})\right)\Big] \begin{bmatrix} \mathbf{c}_{\iota_K(1)} \\ \vdots \\ \mathbf{c}_{\iota_K(n_d+1)} \end{bmatrix} - \mathbf{g}(\mathbf{z}), \tag{2.3.7}$$

with

$$\mathcal{B}(\phi(\mathbf{z})) := \sum_{i=1}^{n_d} \frac{\partial}{\partial x_i}\phi(\mathbf{z})\mathbf{B}_i + \phi\mathbf{B}_0\ . \tag{2.3.8}$$

Note that in our implementation, all collocation points within the domain are classified into unique elements according to certain algorithms to ensure the above evaluation is only done once for all these points; we refer readers to the **findElement** function in the implementation code **Collocation1D** and **Collocation2D** in Appendix B.1 for more detail.

Define a slice function $s_m(i) := i(m-1)+1 : im$, and

$$\mathbf{G}_\Omega^K(\mathbf{z}) \ := \ \Big[\ \mathcal{A}\left(\phi_1^K(\mathbf{z})\right) \quad \cdots \quad \mathcal{A}\left(\phi_{n_d+1}^K(\mathbf{z})\right)\Big] \in \mathbb{R}^{m_\Omega \times n_u(n_d+1)}, \tag{2.3.9}$$

$$\mathbf{G}_\Gamma^K(\mathbf{z}) \ := \ \Big[\ \Lambda\mathcal{B}\left(\phi_1^K(\mathbf{z})\right) \quad \cdots \quad \Lambda\mathcal{B}\left(\phi_{n_d+1}^K(\mathbf{z})\right)\Big] \in \mathbb{R}^{m_\Gamma \times n_u(n_d+1)}. \tag{2.3.10}$$

For all collocation points $\{\mathbf{z}_i\}_{i=1}^{N_\Omega} \subset \Omega$ and $\{\hat{\mathbf{z}}_j\}_{j=1}^{N_\Gamma} \subset \Gamma$, assigning

$$\mathbf{G}_\Omega(\ s_{m_\Omega}(i)\ ,\ s_{n_u} \circ \iota_K(1:n_d+1)\ ) := \mathbf{G}_\Omega^K(\mathbf{z}_i), \tag{2.3.11}$$

$$\mathbf{G}_\Gamma(\ s_{m_\Gamma}(j)\ ,\ s_{n_u} \circ \iota_K(1:n_d+1)\ ) := \mathbf{G}_\Gamma^K(\hat{\mathbf{z}}_j), \tag{2.3.12}$$

$$\mathbf{d}_\Omega(\ s_{m_\Omega}(i)\ ) := \mathbf{f}(\mathbf{z}_i), \tag{2.3.13}$$

$$\mathbf{d}_\Gamma(\ s_{m_\Gamma}(j)\ ) := \Lambda\mathbf{g}(\hat{\mathbf{z}}_j), \tag{2.3.14}$$

we could then obtain a discrete least-squares problem: find $\mathbf{c}^*$ such that

$$\mathbf{c}^* = \underset{\mathbf{c}\in\mathbb{R}^{n_u n_v}}{\operatorname{argmin}} \|\mathbf{G}\mathbf{c} - \mathbf{d}\|_2^2 \tag{2.3.15}$$

where

$$\mathbf{G} = \begin{bmatrix} \mathbf{G}_\Omega \\ \mathbf{G}_\Gamma \end{bmatrix} \quad , \quad \mathbf{c} = \operatorname{vec}(\mathbf{C}) \quad , \quad \mathbf{d} = \begin{bmatrix} \mathbf{d}_\Omega \\ \mathbf{d}_\Gamma \end{bmatrix}. \tag{2.3.16}$$

For the sake of clarity, in formulas (2.3.11) and (2.3.12), $s_{n_u} \circ \iota_K(1:n_d+1)$ is used instead of $[s_{n_u} \circ \iota_K(1), \cdots, s_{n_u} \circ \iota_K(n_d+1)]$ with a slight abuse of notation.

The dimension of the coefficient matrix $\mathbf{G}$ is $(m_\Omega N_\Omega + m_\Gamma N_\Gamma) \times (n_u n_v)$, and when a sufficient number of collocation points are selected to form an over-determined system, minimizing the discrete residual error (2.1.10) is equivalent to solving the normal equation

$$\mathbf{G}^{\mathrm{T}}\mathbf{G}\mathbf{c}^* = \mathbf{G}^{\mathrm{T}}\mathbf{d}. \tag{2.3.17}$$

Procedures given in (2.3.5) – (2.3.17) are summarised in the following Algorithm 1. The below naive algorithm, however, is not practical for the assembly of extremely large problems where the scale of $\mathbf{G} \in \mathbb{R}^{m_\mathbf{G} \times n_\mathbf{G}}$ could reach at least $\mathcal{O}(10^5)$ for

$m_{\mathbf{G}}$ and $\mathcal{O}(10^4)$ for $n_{\mathbf{G}}$ considering the limited memory space of an ordinary laptop. Therefore, in practical implementation, it is necessary to take into account the sparse structure of $\mathbf{G}$. Again, we refer to the implementation code **Collocation1D** and **Collocation2D** in Appendix B.1 for more detail.

**Algorithm 1:** Assembly Algorithm for LSFECM

```
Initialize: $\mathbf{G}_\Omega$ , $\mathbf{d}_\Omega$ $\longleftarrow$ $O_{m_\Omega N_\Omega \times n_u n_v}$ , $O_{m_\Omega N_\Omega \times 1}$ ;
Initialize: $\mathbf{G}_\Gamma$ , $\mathbf{d}_\Gamma$ $\longleftarrow$ $O_{m_\Gamma N_\Gamma \times n_u n_v}$ , $O_{m_\Gamma N_\Gamma \times 1}$ ;
for $\mathbf{z}_i \in \{\mathbf{z}_i\}_{i=1}^{N_\Omega}$ do
    find the element $K \in \mathcal{M}_h$ where $\mathbf{z}_i$ belongs ;
    evaluate $\mathbf{G}_\Omega^K(\mathbf{z}_i)$ with formula (2.3.9) ;
    $\mathbf{i} \longleftarrow s_{m_\Omega}(i)$ ; $\mathbf{j} \longleftarrow s_{n_u} \circ \iota_K(1 : n_d + 1)$ ;
    $\mathbf{G}_\Omega(\mathbf{i}, \mathbf{j}) \longleftarrow \mathbf{G}_\Omega^K(\mathbf{z}_i)$ ; $\mathbf{d}_\Omega(\mathbf{i}) \longleftarrow \mathbf{f}(\mathbf{z}_i)$ ;
for $\hat{\mathbf{z}}_j \in \{\hat{\mathbf{z}}_j\}_{j=1}^{N_\Gamma}$ do
    find the element $K \in \mathcal{M}_h$ where $\hat{\mathbf{z}}_j$ belongs ;
    evaluate $\mathbf{G}_\Gamma^K(\hat{\mathbf{z}}_j)$ with formula (2.3.10) ;
    $\mathbf{i} \longleftarrow s_{m_\Gamma}(j)$ ; $\mathbf{j} \longleftarrow s_{n_u} \circ \iota_K(1 : n_d + 1)$ ;
    $\mathbf{G}_\Gamma(\mathbf{i}, \mathbf{j}) \longleftarrow \mathbf{G}_\Gamma^K(\hat{\mathbf{z}}_j)$ ; $\mathbf{d}_\Gamma(\mathbf{i}) \longleftarrow \Lambda\mathbf{g}(\hat{\mathbf{z}}_j)$ ;
$\mathbf{G} \longleftarrow [\mathbf{G}_\Omega \ ; \ \mathbf{G}_\Omega]$ ; $\mathbf{d} \longleftarrow [\mathbf{d}_\Omega \ ; \ \mathbf{d}_\Omega]$ ;
```

It is also worth noting that a more natural and practical choice is to iterate over the cells $K \in \mathcal{M}_h$ rather than over the collocation points. The above Algorithm 1 is just given in a form consistent with our actual implementation.

### 2.3.2 Implementation in 1D

Consider solving the boundary value problem in 1D

$$\mathbf{A}_1\mathbf{u}'(x) + \mathbf{A}_0\mathbf{u}(x) = \mathbf{f}(x) \quad \text{in} \quad \Omega = [a, b], \tag{2.3.18a}$$

$$\mathbf{B}_1\mathbf{u}'(x) + \mathbf{B}_0\mathbf{u}(x) = \mathbf{g}(x) \quad \text{in} \quad \Gamma = \{a, b\}. \tag{2.3.18b}$$

The domain $\Omega$ is discretised by by partitioning with $n_v$ (distinct) global vertices $\{v_i\}_{i=1}^{n_v}$, where $a = v_1 < v_2 < \cdots < v_{n_v} = b$, and forming elements by letting $K_i = [v_i, v_{i+1}]$. In each element $K \in \mathcal{M}_h$ of the mesh $\mathcal{M}_h = \{K_i\}_{i=1}^{n_v - 1}$, where

$K = [v_1^K, v_2^K]$, the nodal bases for CG$_1$ elements in 1D are just

$$\phi_1^K(x) \;=\; \frac{v_2^K - x}{L^K} \quad \text{and} \quad \phi_2^K(x) \;=\; \frac{x - v_1^K}{L^K}, \tag{2.3.19}$$

with $L^K = v_2^K - v_1^K$ representing the length of the element $K$. For the $ith$ element $K_i$, the corresponding local-to-global map $\iota_{K_i}(j) : [2] \mapsto [n_v]$ is given by $\iota_{K_i}(j) = i+(j-1)$. The following plots in Fig.2.3.1 display the bases for 1D CG$_1$ elements.

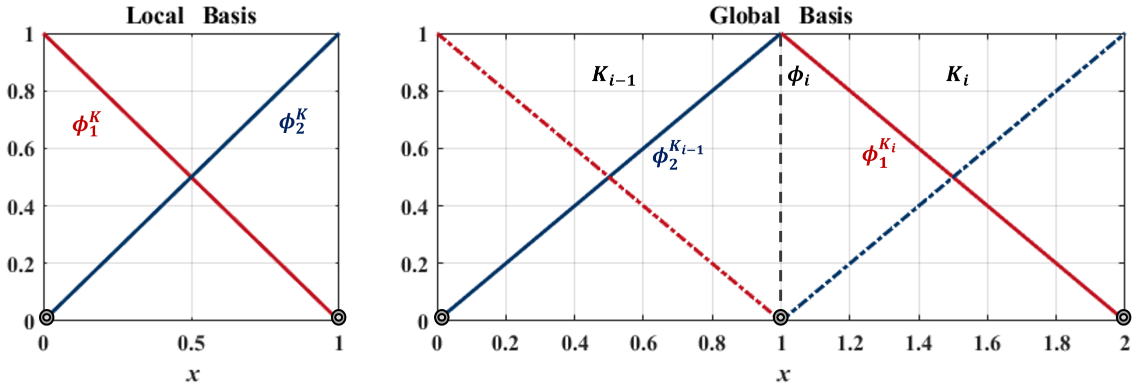

**Fig.2.3.1** Local Bases for the 1D CG$_1$ element over the reference interval $[0, 1]$ (left); and the assembly to global basis (right).

For the evaluation of residuals, we have

$$\mathcal{A}\left(\phi_1^K\right) \;=\; -\frac{1}{L^K}\mathbf{A}_1 + \phi_1^K\mathbf{A}_0 \quad , \quad \mathcal{A}\left(\phi_2^K\right) \;=\; \frac{1}{L^K}\mathbf{A}_1 + \phi_2^K\mathbf{A}_0, \tag{2.3.20}$$

$$\mathcal{B}\left(\phi_1^K\right) \;=\; -\frac{1}{L^K}\mathbf{B}_1 + \phi_1^K\mathbf{B}_0 \quad , \quad \mathcal{B}\left(\phi_2^K\right) \;=\; \frac{1}{L^K}\mathbf{B}_1 + \phi_2^K\mathbf{B}_0. \tag{2.3.21}$$

Unless otherwise specified, for a mesh of size $h$, $\mathcal{O}(h^{-2})$ equally spaced collocation points will be chosen to fit the interior condition, while only two endpoints will be used to evaluate the boundary residual with respect to a weight matrix of $\mathbf{\Lambda} = h^{-1.5}\,\mathbf{I}_{m_\Gamma}$, where $\mathbf{I}_{m_\Gamma}$ represents the identity matrix of size $m_\Gamma \times m_\Gamma$. The reason for choosing these specific parameters will be discussed in detail in Chapter 5.

The example problem we consider here for the 1D formulation is given by setting

$$\begin{bmatrix} \cos(x) & -\sin(x) \\ \sin(x) & \cos(x) \end{bmatrix} \begin{bmatrix} u_1' \\ u_2' \end{bmatrix} + \begin{bmatrix} \cos(x) & -\sin(x) \\ \sin(x) & \cos(x) \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}, \tag{2.3.22a}$$

$$\begin{bmatrix} 1 & 1 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} = \mathbb{I}_0(x) - \mathbb{I}_\pi(x), \tag{2.3.22b}$$

over the domain $\Omega = [0, \pi]$ and boundary $\Gamma = \{0, \pi\}$ respectively. Here $\mathbb{I}_a(x)$ denotes the indicator function such that $\mathbb{I}_a(a) = 1$ and $\mathbb{I}_a(x) = 0$ otherwise. The solution to the problem is simply

$$\mathbf{u} = \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} = \begin{bmatrix} \sin(x) \\ \cos(x) \end{bmatrix}. \tag{2.3.23}$$

Fig.2.3.2 visualises a simple example for the coefficient matrix $\mathbf{G}$ of the assembled algrbraic system $\mathbf{G}^{\mathrm{T}}\mathbf{G}\mathbf{c}^{*} = \mathbf{G}^{\mathrm{T}}\mathbf{d}$, with $N_{\Omega} = 20$ and $N_{\Gamma} = 2$ in 1D. The final dimension of the system is $42 \times 20$. It could be observed that for a set of well-ordered collocation points, $\mathbf{G}$ in 1D is both sparse and almost block-diagonal.

For illustration, a numerical solution $\mathbf{u}_h$ to the problem (2.3.22) obtained by discretizing the domain into 999 equal-length elements ($n_v = 1000$) and solving the complete discrete system has been present in Fig.2.3.3 with point-wise absolute error $|\mathbf{u} - \mathbf{u}_h|$ displayed in a $\log_{10}$ scale.

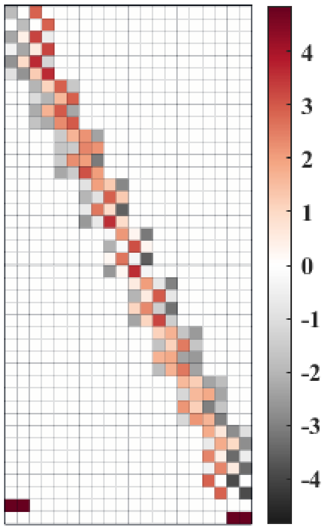

**Fig.2.3.2** Coefficient matrix $\mathbf{G}$ in 1D formulation.

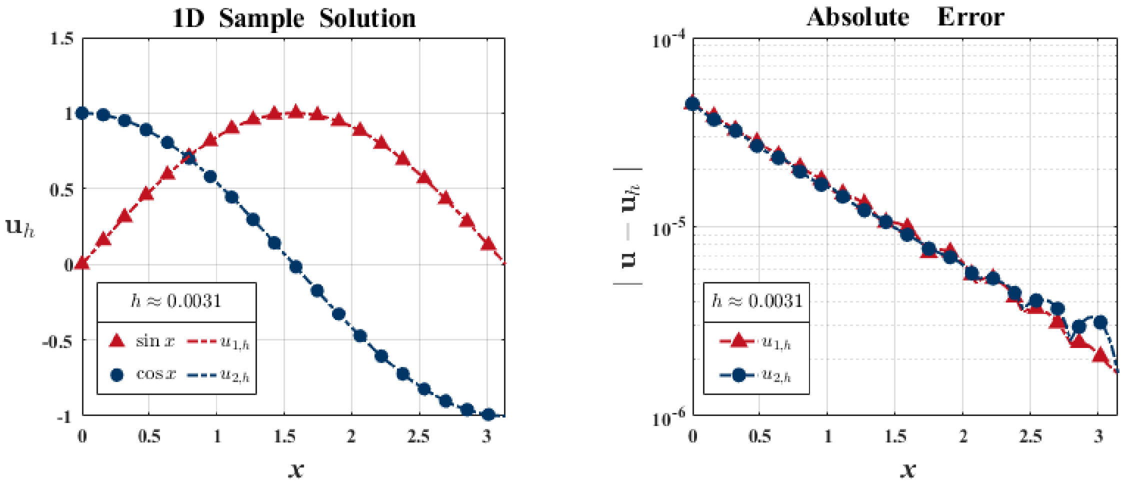

**Fig.2.3.3** Numerical solution to the example problem obtained by using LSFECM and solving the complete over-determined system (left); point-wise absolute error displayed in $\log_{10}$ scale (right).

### 2.3.3 Implementation in 2D

We now consider a slightly more complicated implementation where the boundary value problems are solved in 2D, namely, for $\mathbf{u}(x,y)$ defined over some Lipschitz domain $\Omega$, solving

$$\mathbf{A}_1\mathbf{u}_x + \mathbf{A}_2\mathbf{u}_y + \mathbf{A}_0\mathbf{u} = \mathbf{f}(x,y) \quad \text{in} \quad \Omega, \tag{2.3.24a}$$

$$\mathbf{B}_1\mathbf{u}_x + \mathbf{B}_2\mathbf{u}_y + \mathbf{B}_0\mathbf{u} = \mathbf{g}(x,y) \quad \text{in} \quad \Gamma = \partial\Omega, \tag{2.3.24b}$$

where we use subscript to denote partial derivatives. The 2D domains are discretised using the function **generateMesh** in MATLAB's Partial Differential Equation Toolbox [22]. The output of the function will contain a set of global vertices $\mathbb{V} = \{\mathbf{v}_i\}_{i=1}^{n_v}$ as well as a local-to-global map $\{\iota_K\}_{K\in\mathcal{M}_h}$ for each element $K = \triangle(\mathbf{v}_1^K, \mathbf{v}_2^K, \mathbf{v}_3^K)$, which is a triangle where vertices $\mathbf{v}_1^K, \mathbf{v}_2^K, \mathbf{v}_3^K$ are given in counterclockwise order. The scalar nodal bases for each element $K$ and $\mathbf{x} \in K$ are now given by

$$\phi_1^K(\mathbf{x}) = \frac{S_\triangle(\mathbf{x}, \mathbf{v}_2^K, \mathbf{v}_3^K)}{S^K}, \tag{2.3.25}$$

$$\phi_2^K(\mathbf{x}) = \frac{S_\triangle(\mathbf{v}_1^K, \mathbf{x}, \mathbf{v}_3^K)}{S^K}, \tag{2.3.26}$$

$$\phi_3^K(\mathbf{x}) = \frac{S_\triangle(\mathbf{v}_1^K, \mathbf{v}_2^K, \mathbf{x})}{S^K}, \tag{2.3.27}$$

where

$$S_\triangle(\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3) = \frac{1}{2}\begin{vmatrix} 1 & x_1 & y_1 \\ 1 & x_2 & y_2 \\ 1 & x_3 & y_3 \end{vmatrix} \tag{2.3.28}$$

denotes the area of the triangle $\triangle(\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3)$ with vertices $\mathbf{x}_i = (x_i, y_i)$, and $S^K = S_\triangle(\mathbf{v}_1^K, \mathbf{v}_2^K, \mathbf{v}_3^K)$ is the area of the triangular element $K$. In Fig.2.3.4, we provide a visualisation for 2D CG$_1$ bases over the reference triangle $\triangle((0,1),(0,0),(1,0))$ as well as the assembled global basis.

The sampling procedure here is slightly more complicated compared to that in 1D. The boundary collocation points can either be chosen directly as part of the global vertices $\{\mathbf{v}_i \in \mathbb{V} : \mathbf{v}_i \in \Gamma\}$, or can be chosen in an "equally spaced" manner over distinct boundaries, such that the segments between two adjacent collocation points are of equal length. Interior collocation points are sampled "uniformly" from each triangular element $K = \triangle(\mathbf{v}_1^K, \mathbf{v}_2^K, \mathbf{v}_3^K)$ in equal amounts using the formula below [35],

$$\mathbf{x}_i = (1 - \sqrt{r_1})\mathbf{v}_1^K + \sqrt{r_1}(1 - r_2)\mathbf{v}_2^K + \sqrt{r_1}r_2\mathbf{v}_3^K, \tag{2.3.29}$$

where $r_1$ and $r_2$ are two independent uniform random variables within the range $[0, 1]$. For a triangular mesh $\mathcal{M}_h$ of size $h$, we will, by default, choose at least $\mathcal{O}(h^{-4})$ interior collocation points and $\mathcal{O}(h^{-2})$ equally spaced collocation points along each boundary of the domain.

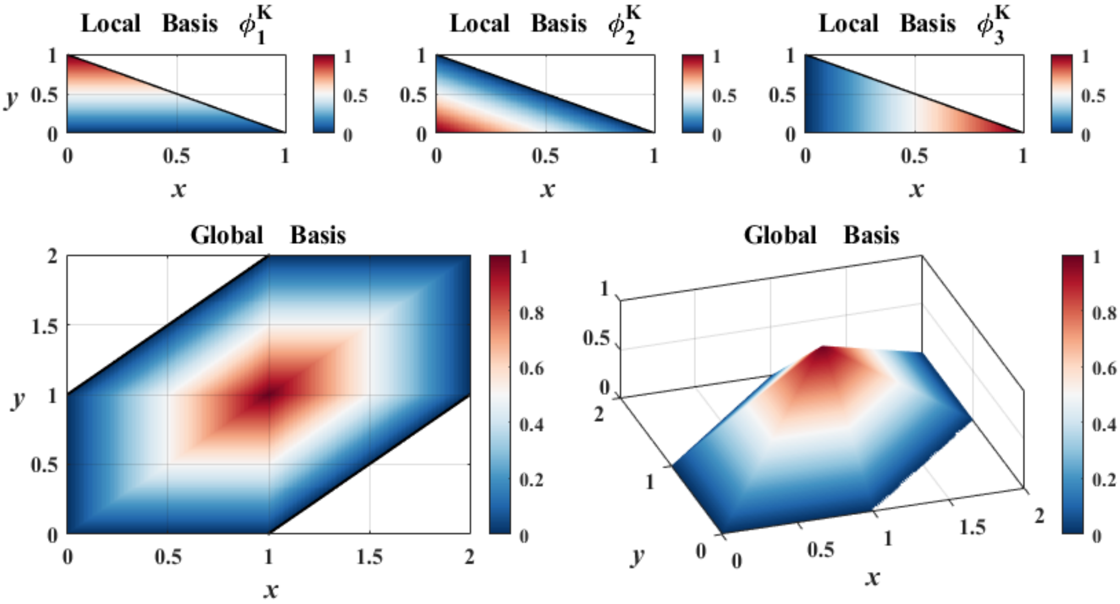

**Fig.2.3.4** Local Bases for the 2D CG$_1$ element over the reference triangle (above); and the assembled global basis (below).

For a computational purpose, we could evaluate $\mathbf{G}_\Omega^K(\mathbf{z})$ and $\mathbf{G}_\Gamma^K(\mathbf{z})$ using the following expressions with a slight re-arrangement on the original formula (2.3.9) and (2.3.10),

$$\mathbf{G}_\Omega^K(\mathbf{z}) = \begin{bmatrix}\mathbf{A_1} & \mathbf{A_2} & \mathbf{A_0}\end{bmatrix} \begin{bmatrix} \partial_x\phi_1^K & \partial_x\phi_2^K & \partial_x\phi_3^K \\ \partial_y\phi_1^K & \partial_y\phi_2^K & \partial_y\phi_3^K \\ \phi_1^K & \phi_2^K & \phi_3^K \end{bmatrix} \otimes \mathbf{I}_{n_u}, \tag{2.3.30}$$

and

$$\mathbf{G}_\Gamma^K(\mathbf{z}) = \Lambda \begin{bmatrix}\mathbf{B_1} & \mathbf{B_2} & \mathbf{B_0}\end{bmatrix} \begin{bmatrix} \partial_x\phi_1^K & \partial_x\phi_2^K & \partial_x\phi_3^K \\ \partial_y\phi_1^K & \partial_y\phi_2^K & \partial_y\phi_3^K \\ \phi_1^K & \phi_2^K & \phi_3^K \end{bmatrix} \otimes \mathbf{I}_{n_u}, \tag{2.3.31}$$

where $\otimes$ represents the Kronecker tensor product. The derivatives in the above expressions can be calculated exactly as they are just the corresponding coefficients of terms in a linear function. The weight matrix for boundary residuals here is by default $\mathbf{\Lambda} = h^{-1.5}\mathbf{I}_{m_\Gamma}$.

The specific problem we consider here in 2D is a Poisson equation with inhomogeneous boundary conditions. The decomposition process for this kind of problem has already been demonstrated in Section 2.2. Adhering to the previous notations (from Section 2.2) and rewriting the equations (2.2.2) in matrix form, the problem to solve then becomes

$$\begin{bmatrix}1 & 0 & 0\\ 0 & 0 & 0\\ 0 & -1 & 0\\ 0 & 0 & 1\end{bmatrix}\begin{bmatrix}\varphi_x\\ w_{1x}\\ w_{2x}\end{bmatrix} + \begin{bmatrix}0 & 0 & 0\\ 1 & 0 & 0\\ 0 & 0 & -1\\ 0 & -1 & 0\end{bmatrix}\begin{bmatrix}\varphi_y\\ w_{1y}\\ w_{2y}\end{bmatrix} + \begin{bmatrix}0 & -1 & 0\\ 0 & 0 & -1\\ 0 & 0 & 0\\ 0 & 0 & 0\end{bmatrix}\begin{bmatrix}\varphi\\ w_1\\ w_2\end{bmatrix} = \begin{bmatrix}0\\ 0\\ f\\ 0\end{bmatrix} \quad \text{in} \quad \Omega, \tag{2.3.32a}$$

$$\begin{bmatrix}1 & 0 & 0\\ 0 & n_2 & -n_1\end{bmatrix}\begin{bmatrix}\varphi\\ w_1\\ w_2\end{bmatrix} = \begin{bmatrix}g\\ \nabla g \times \mathbf{n}\end{bmatrix} \quad \text{in} \quad \Gamma, \tag{2.3.32b}$$

where we use $\nabla\varphi = \mathbf{w} = [w_1, w_2]^{\mathrm{T}}$ and $\mathbf{n} = [n_1, n_2]^{\mathrm{T}}$. For a more concrete example, we consider the above boundary value problem (2.3.32) with an exact solution

$$\mathbf{u} = \begin{bmatrix}\varphi\\ w_1\\ w_2\end{bmatrix} = \begin{bmatrix}\sin(x^2+y^2)\\ 2x\cos(x^2+y^2)\\ 2y\cos(x^2+y^2)\end{bmatrix}. \tag{2.3.33}$$

Therefore, we set

$$f = 4(x^2+y^2)\sin(x^2+y^2) - 4\cos(x^2+y^2), \tag{2.3.34}$$

$$g = \sin(x^2+y^2)|_\Gamma. \tag{2.3.35}$$

The example problem is considered over the polygonal approximation $\Omega_h$ of the domain $\Omega$, which contains three holes.

$$\begin{aligned}\Omega = \{\ (x,y)\ :\ & x^2+y^2 \le 2.25,\\ & (x-0.5)^2+y^2 \ge 0.25,\\ & (x-0.5)^2+(y-1) \ge 0.04,\\ & (x+0.6)^2+(y+0.8)^2 \ge 0.09\ \}.\end{aligned} \tag{2.3.36}$$

A numerical solution $\mathbf{u}_h$ to the example problem obtained by solving the complete discrete system has been demonstrated in Fig.2.3.5, together with the corresponding coefficient matrix $\mathbf{G}$ with a row-column ratio approximately equal to 18.67, generated by 5040 interior and 199 boundary collocation points. The mesh size was $h = 0.15$, leading to a total number of 1101 unknowns. Once again, we can see from Fig.2.3.5 that $\mathbf{G}$ is sparse. Yet, this time, the matrix is no longer (almost) block-diagonal.

The $L^2$ error for $\mathbf{u}_h$, is calculated to be approximately $\|\mathbf{u}_h - \mathbf{u}\|_{0,\Omega_h} \approx 0.0533$, indicating a relatively accurate approximation to the true solution (2.3.33). The estimation for the error integral over the domain was obtained by first approximating the value over each element of the mesh using the 4-point Gaussian quadrature for triangles and then summing up all those evaluations,

$$\begin{aligned} \|\mathbf{u}_h - \mathbf{u}\|_{0,\Omega_h}^2 = \int_{\Omega_h} \|\mathbf{u}_h - \mathbf{u}\|_2^2 \, \mathrm{d}\mathbf{x} = \sum_{K\in\mathcal{M}_h} \int_K \|\mathbf{u}_h - \mathbf{u}\|_2^2 \, \mathrm{d}\mathbf{x} \\ \approx \sum_{K\in\mathcal{M}_h} \sum_{i=1}^{4} \omega_i^K \|\mathbf{u}_h(\mathbf{p}_i^K) - \mathbf{u}(\mathbf{p}_i^K)\|_2^2, \end{aligned} \tag{2.3.37}$$

where $\mathcal{M}_h$ is the triangular mesh for $\Omega_h$, $\omega_i^K$ and $\mathbf{p}_i^K$ are the corresponding weights and Gaussian points to perform the Gaussian quadrature over the triangular element $K$; see [40].

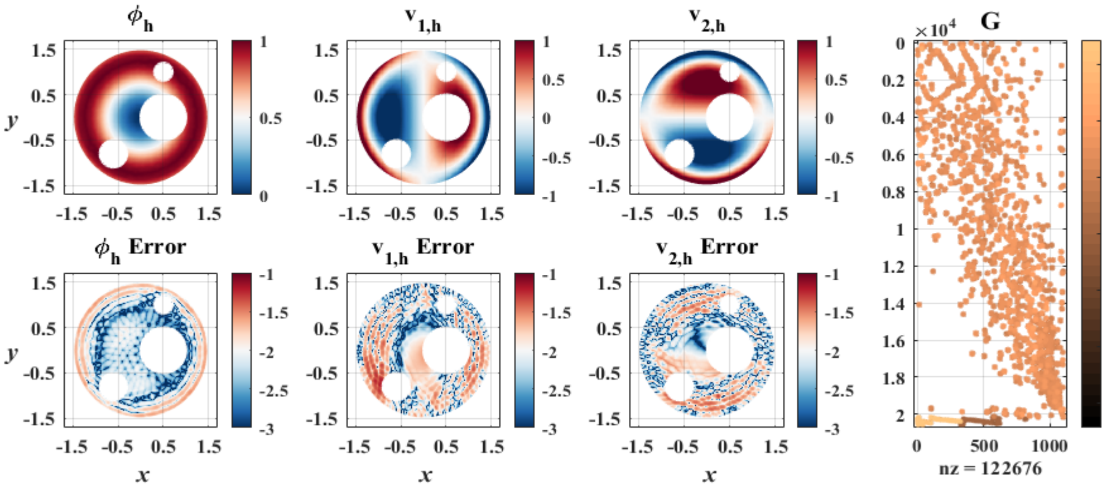

**Fig.2.3.5** Numerical solution to the 2D Poisson equation obtained by using LSFECM and solving the complete over-determined system (above), point-wise absolute error displayed in log10 scale (below), and the corresponding coefficient matrix $\mathbf{G}$ in the 2D formulation (right).

# Chapter 3

# Sketch-and-Solve Methods

As has been discussed in Chapter 2, the LSFECM will eventually lead to a sparse linear least-squares problem (2.3.15), where the row dimension of the coefficient matrix $\mathbf{G}$ is proportional to the number of collocation points considered. Hence, theoretically speaking, problem (2.3.15) can be made arbitrarily over-determined to achieve the desired accuracy. The above characterization of LSFECM has led to the natural idea of whether it is possible to reduce the dimension of the final over-determined system by sketching, thus speeding up the solving process while also preserving the accuracy of the obtained solution.

Based on the thoughts above, this chapter provides a detailed introduction to the sketching methods, i.e., subspace embeddings, which will be used to solve the discrete linear least-squares problem (2.3.15) derived from the LSFECM later in Chapter 4.

## 3.1 Preliminaries

The core idea of the sketch-and-solve method is simple: replace the original over-determined system with a new one with fewer but well-mixed equations. A more formal definition is given as follows.

**Definition 3.1.1. Sketch-and-Solve Methods**
*For a linear least-squares problem with coefficient matrix $\mathbf{A} \in \mathbb{R}^{m\times n}$, $m \gg n$ and*

$$\mathbf{x}^* = \min_{\mathbf{x}\in\mathbb{R}^n} \|\mathbf{A}\mathbf{x} - \mathbf{b}\|_2, \tag{3.1.1}$$

*the corresponding sketched solution with respect to some pre-defined sketch $\mathbf{S} \in \mathbb{R}^{d\times m}$,*

*$n < d < m$, is given by*

$$\hat{\mathbf{x}} = \min_{\mathbf{x} \in \mathbb{R}^n} \|\mathbf{S}(\mathbf{A}\mathbf{x} - \mathbf{b})\|_2, \tag{3.1.2}$$

*where we assume both the original and the sketched coefficient matrices are full rank* $\operatorname{rank}(\mathbf{A}) = \operatorname{rank}(\mathbf{SA}) = n$.

Solving the normal equation of the linear least-squares problem (3.1.1) via the QR decomposition usually admits a complexity of $\mathcal{O}(mn^2)$. The solution to the sketched problem (3.1.2) can be obtained under a complexity of $\mathcal{O}(dn^2)$ using the same technique. However, the general complexity of the sketch-and-solve method is also bounded by the number of operations needed to apply the sketch $\mathbf{S}$, leading to a total complexity of $\mathcal{O}(dmn + dn^2)$, for $d > n$. We therefore conclude that the method is only practical if the complexity of sketching can be controlled below $\mathcal{O}(mn^2)$.

Reducing the complexity of sketching is often achieved via applying fast transforms such as the Hadamard transform in [2] and [16, §2], the fast Fourier transform and its variants in [2], or simply by applying sparse sketches such as Hashing matrices [9, §2] or leverage score samplers [29]. We shall describe some of these implementations in detail in the next section, before that, the following definitions are made.

**Definition 3.1.2. Subspace Embeddings [9, §2] [16, §2] [41, §2]**
*A matrix $\mathbf{S} \in \mathbb{R}^{d \times m}$ is an $\epsilon$-Johnson-Lindenstrauss transform* ($\epsilon$-JLT) *for some distortion $\epsilon \in (0, 1)$, and a subset $\mathbb{Y} \subset \mathbb{R}^m$ if*

$$(1 - \epsilon)\|\mathbf{y}\|_2^2 \leq \|\mathbf{Sy}\|_2^2 \leq (1 + \epsilon)\|\mathbf{y}\|_2^2 \quad \text{for all} \quad \mathbf{y} \in \mathbb{Y}. \tag{3.1.3}$$

$\mathbf{S}$ *is called an $\epsilon$-subspace embedding* ($\epsilon$-SE) *for the given matrix $\mathbf{A} \in \mathbb{R}^{m \times n}$ if it is an* $\epsilon$-JLT *for the column space of $\mathbf{A}$. If $\mathbf{S}$ forms an* $\epsilon$-SE *for $\mathbf{A}$, the rank is retained, i.e.,* $\operatorname{rank}(\mathbf{SA}) = \operatorname{rank}(\mathbf{A})$*; see* [37, §2].

**Definition 3.1.3. The Fast Johnson-Lindenstrauss Transform [16, §2]**
$\mathbf{S} \in \mathbb{R}^{d \times m}$ *is a fast $\epsilon$-Johnson-Lindenstrauss transform* ($\epsilon$-FJLT) *for the given matrix $\mathbf{A} \in \mathbb{R}^{m \times n}$ if it is an* $\epsilon$-SE *for $\mathbf{A}$, and the matrix product $\mathbf{SA}$ can be computed in $\mathcal{O}(mn \ln(d))$ flops.*

One simple way to construct an $\epsilon$-JLT for a finite set of vectors $\{\mathbf{y}_i\}_{i=1}^n \subset \mathbb{R}^m$ was proposed by Achlioptas in [1]. We summarized his construction in the following Theorem 3.1.1, which will be used later in Section 3.2.2 to help compute fast estimations of leverage scores [16, §3].

**Theorem 3.1.1.** [1] [16, §2] *Given a finite set of vectors* $\{\mathbf{y}_i\}_{i=1}^n \subset \mathbb{R}^m$*, and let the elements in* $\mathbf{S} \in \mathbb{R}^{d\times m}$ *be i.i.d. random variables following the distribution* $\mathbf{p} = (p_1, p_2, p_3)$ *such that*

$$\mathbf{S}_{i,j} = \begin{cases} +\sqrt{3/d} & \text{with } p_1 = 1/6, \\ -\sqrt{3/d} & \text{with } p_2 = 1/6, \\ 0 & \text{with } p_3 = 2/3. \end{cases} \tag{3.1.4}$$

*Then,* $\mathbf{S}$ *can produce an* $\epsilon$-JLT *for* $\{\mathbf{y}_i\}_{i=1}^n$ *with high probability for some* $\epsilon \in (0, 0.5]$ *and* $d \sim \underline{\Omega}(\epsilon^{-2}\ln(n))$*.*

We also have the following classification for different strategies to construct subspace embeddings.

**Definition 3.1.4. (Non-)Oblivious Embeddings [9, §2] [41, §2]**
*A distribution* $\mathcal{S}$ *is an* $(\epsilon, \delta)$*-oblivious subspace embedding, or* $(\epsilon, \delta)$-OSE*, if the sketch* $\mathbf{S}$ *drawn from the distribution forms an* $\epsilon$-SE *for any matrices* $\mathbf{A}$ *with probability at least* $1 - \delta$*. A distribution* $\mathcal{S}_\mathbf{A}$ *for some fixed matrix* $\mathbf{A}$ *is an* $(\epsilon, \delta)$*-non-oblivious subspace embedding, or* $(\epsilon, \delta)$-NOSE*, if the sketch* $\mathbf{S}$ *drawn from the distribution forms an* $\epsilon$-SE *for* $\mathbf{A}$ *with probability at least* $1 - \delta$*.*

The oblivious subspace embeddings used here are mainly Gaussian projections [31, §8], and Hashing embeddings [9, §2]. For non-oblivious subspace embeddings, we consider the leverage score sampling and its approximations [16, §3] [29].

## 3.2 Subspace Embeddings

### 3.2.1 Oblivious Subspace Embeddings

Here, we describe some of the oblivious subspace embeddings that may be used later in our implementation. One of the most commonly mentioned methods to construct such embeddings is given by the Gaussian projection.

**Definition 3.2.1. Gaussian Projection [31, §8] [39, §3]**
*Let* $\mathcal{N}(\mu, \sigma^2)$ *denote the normal distribution with mean* $\mu$ *and variance* $\sigma^2$*. A Gaussian projection matrix* $\mathbf{S} \in \mathbb{R}^{d\times m}$ *is defined as*

$$\mathbf{S} = \frac{1}{\sqrt{d}}\mathbf{N} \tag{3.2.1}$$

*where each element in* $\mathbf{N}$ *are i.i.d. standard normal random variables* $\mathbf{N}_{ij} \sim \mathcal{N}(0, 1)$*.*

As a result of the spectral distribution of Gaussian matrices combined with their rotational invariance [11, §1] [31, §8] [36], the Gaussian projection can provide an $(\epsilon, \delta)$-OSEs if $d \sim \Theta(\epsilon^{-2} n + \epsilon^{-2} \ln(1/\delta))$; see [41, §6]. However, the method is usually impractical due to the high time complexity to perform the matrix multiplication. Due to its dense nature, the number of operations needed to calculate $\mathbf{SA}$ is usually $\mathcal{O}(dmn)$ for some $\mathbf{A} \in \mathbb{R}^{m \times n}$ and $\mathbf{S}$ in (3.2.1). Furthermore, applying the Gaussian projection will also break the sparsity of the original matrix. Therefore, we propose the sHashing embedding [9, §2], which provides a more efficient alternative to Gaussian projections.

**Definition 3.2.2. s-Hashing Embedding [9, §2] [37, §1]**
$\mathbf{S} \in \mathbb{R}^{d \times m}$ *is a s-hashing matrix, if for each column* $j \in [m]$, $s$ *rows* $i_1, i_2, \cdots, i_s \in [d]$ *are selected uniformly at random without replacement so that*

$$\mathbf{S}_{i,j} = \begin{cases} \pm 1/\sqrt{s} & \text{if } i \in \{i_k\}_{k=1}^{s} \\ 0 & \text{otherwise} \end{cases} \tag{3.2.2}$$

*where the signs for* $\mathbf{S}_{i_k,j}$ *are determined with equal probability. Sketching with the 1-hashing matrix is usually referred to as the* ***Count sketch*** *[39].*

In [14], Cohen shows that sketching via s-hashing matrices with $d \sim \mathcal{O}(\epsilon^{-2} n \ln(n/\delta))$ and $s \sim \mathcal{O}(\epsilon^{-2} \ln(n/\delta))$ for $\mathbf{A} \in \mathbb{R}^{m \times n}$ can admit $(\epsilon, \delta)$-OSEs while the complexity for matrix multiplication is only $\mathcal{O}(mn \ln(n))$. For the Count sketch, the complexity is further reduced to $\mathcal{O}(mn)$; however, the reduction in complexity comes at the cost of increasing the number of sketched rows: $d \sim \mathcal{O}(\epsilon^{-2} n^2)$ rows are needed to construct an OSE with high probability using Count sketch; see [13] and [39, §3].

Here we also outline the following subsampled fast trigonometric transform (SRFT) in [2], [31, §9] and [34, §2], which will be considered later to construct fast Johnson-Lindenstrauss transforms in the next section.

**Definition 3.2.3. Subsampled Fast Trigonometric Transforms [31, §9]**
$\mathbf{S} \in \mathbb{R}^{d \times m}$ *is a subsampled fast trigonometric transform* SRFT*, if*

$$\mathbf{S} = \mathbf{RFD}, \tag{3.2.3}$$

*where* $\mathbf{D}$ *is a diagonal matrix of independent Rademacher variables,* $\mathbf{F} \in \mathbb{R}^{m \times m}$ *is a fast unitary trigonometric transform which can be applied to a* $m \times n$ *matrix in* $\mathcal{O}(mn \ln(m))$ *flops and* $\mathbf{R}$ *randomly select* $d$ *rows from a column vector of length* $m$*.*

A common and theoretically guaranteed choice for $\mathbf{F}$ may be the normalised Hadamard transform used in [2], [16, §2], [17] and [41, §2], while it could only be used when $m$ is a power of 2. Other popular alternatives to the Hadamard transform include normalised fast Fourier transforms for complex vectors and normalised discrete cosine transforms for real vectors [2]. In our implementation, normalised discrete cosine transforms will be applied to construct SRFTs (and FJLTs) using MATLAB's DCT-4 variant [21].

### 3.2.2 Non-Oblivious Subspace Embeddings

In addition to the oblivious subspace embeddings introduced in Section 3.2.1, we also consider constructing non-oblivious subspace embeddings using (approximate) leverage scores. We will begin by reviewing the following definitions.

**Definition 3.2.4. Leverage Scores and the Coherence [16, §3] [34, §6]**
*For an $m \times n$ real matrix $\mathbf{A}$ with $\operatorname{rank}(\mathbf{A}) = r$ whose compact* SVD *is given by $\mathbf{A} = \mathbf{U}_r \mathbf{\Sigma}_r \mathbf{V}_r^{\mathrm{T}}$, let $\mathbf{U}_r(i,:)$ denotes the ith row of $\mathbf{U}_r$; the (row) leverage scores of $\mathbf{A}$ are defined as*

$$\zeta_i = \|\mathbf{U}_r(i,:)\|_2^2 \quad \text{for} \quad i \in [m]. \tag{3.2.4}$$

*We call the largest row leverage score of $\mathbf{A}$ its row coherence $\mu(\mathbf{A})$, i.e., $\mu(\mathbf{A}) = \max_{i\in[m]} \zeta_i$. In fact, one can show that for any $\mathbf{U} \in \mathbb{R}^{m\times r}$ satisfying $\operatorname{span}(\mathbf{U}) = \operatorname{span}(\mathbf{A})$ and $\mathbf{U}^{\mathrm{T}}\mathbf{U} = \mathbf{I}_r$, we have $\zeta_i = \|\mathbf{U}(i,:)\|_2^2$ for all $i \in [m]$; see proof for Theorem A.2.2 in Appendix A.2.*

Note that the definition of row leverage scores (we shall omit the prefix "row" unless otherwise specified) is actually basis-independent. Once we obtain these values (or their approximations) for the target matrix $\mathbf{A}$, we can perform the leverage score sampling as below.

**Definition 3.2.5. Leverage Score Sampling [29] [41, §2]**
*Given an $m \times n$ real matrix $\mathbf{A}$, let $\mathbf{p}_\zeta$ be the distribution given by the normalized leverage scores of $\mathbf{A}$*

$$\mathbf{p}_\zeta(k) = \frac{\zeta_k}{\sum_{l\in[m]} \zeta_l} \quad \text{for} \quad k \in [m], \tag{3.2.5}$$

*and $\hat{\mathbf{p}}_\zeta$ be some approximated distribution for $\mathbf{p}_\zeta$. We say $\mathbf{S} \in \mathbb{R}^{d\times m}$ is a leverage score sampler with respect to $\hat{\mathbf{p}}_\zeta$ if one column index $q \in [m]$ is selected independently for each row $i \in [d]$ in $\mathbf{S}$ according to the distribution $\hat{\mathbf{p}}_\zeta$ such that in the ith row,*

$$\mathbf{S}_{i,j} = \begin{cases} 1/\sqrt{d \cdot \hat{\mathbf{p}}_\zeta(q)} & \text{if } j = q \\ 0 & \text{otherwise} \end{cases} \tag{3.2.6}$$

*we call* $\beta = \min_{k\in[m]}\{\hat{\mathbf{p}}_\zeta(k)/\mathbf{p}_\zeta(k)\} \in (0,1]$ *the misestimation factor of the approximate distribution* $\hat{\mathbf{p}}_\zeta$*. Note that the estimation is exact if and only if* $\beta = 1$*.*

It has been proved in Proposition 6.1.1 of [34, §6], that the above leverage score sampler could provide an $(\epsilon, \delta)$-NOSE if $\beta d \sim \underline{\Omega}(\epsilon^{-2} n \ln(n/\delta))$, or to be more specific, if

$$\beta d \geq [(1+\epsilon)\ln(1+\epsilon) - \epsilon]^{-1} n \ln(2n/\delta), \tag{3.2.7}$$

where $[(1+\epsilon)\ln(1+\epsilon)-\epsilon]^{-1}$ could be replaced by $[\ln(4/e)]^{-1}\epsilon^{-2}$ for $\epsilon \in (0,1]$; see a proof for Corollary A.2.1 in Appendix A.2.

The time for sketching using leverage score samplers is only $\mathcal{O}(n^2\ln(n))$ for $\beta \sim \mathcal{O}(1)$ and $d$ satisfying the condition (3.2.7). Yet, the method is still impractical as finding the orthonormal bases for the column space of $\mathbf{A}$ to compute the exact leverage scores using either SVD or QR would cost $\mathcal{O}(mn^2)$ flops. Nevertheless, we can mitigate the situation using approximate leverage scores that are more efficient to calculate [16]. We summarize the main algorithm (Algorithm 1 in [16, §3]) as given below.

**Algorithm 2:** Fast Approximate Leverage Scores [16, §3]

```
Initialize: Target matrix $\mathbf{A} \in \mathbb{R}^{m\times n}$ with $m \gg n$ and rank($\mathbf{A}$) $= r$ ;
Initialize: The left sketch $\mathbf{S}_1 \in \mathbb{R}^{d_1\times m}$ and the right sketch $\mathbf{S}_2 \in \mathbb{R}^{n\times d_2}$ for
            some $d_2 < n < d_1 < m$ ;
1 Compute the compact SVD of $\mathbf{S}_1\mathbf{A}$ such that $\mathbf{S}_1\mathbf{A} = \hat{\mathbf{U}}\hat{\mathbf{\Sigma}}\hat{\mathbf{V}}^{\mathrm{T}}$ ;
2 Compute $\hat{\mathbf{V}}\hat{\mathbf{\Sigma}}^{-1}$ and assign $\hat{\mathbf{R}}^{-1} \longleftarrow \hat{\mathbf{V}}\hat{\mathbf{\Sigma}}^{-1}\mathbf{S}_2(1:r,:)$ ;
3 Assign $\hat{\mathbf{Q}} \longleftarrow \mathbf{A}\hat{\mathbf{R}}^{-1}$ and approximate the $ith$ leverage score as $\|\hat{\mathbf{Q}}(i,:)\|_2^2$ ;
```

The complexities for doing steps 1 to 3 in Algorithm 2 are $\mathcal{O}(mnd_1 + d_1n^2)$, $\mathcal{O}(n^2 + n^2d_2)$, $\mathcal{O}(mnd_2)$, respectively, incurring an overall complexity of $\mathcal{O}(mnd_1 + mnd_2)$. In the original algorithm proposed by Drineas et al. [16, §3], $\mathbf{S}_1$ was chosen to be a $\epsilon$-FJLT constructed by the subsampled randomized Hadamard transform and $\mathbf{S}_2$ was a $\epsilon$-JLT obtained using Theorem 3.1.1 (by taking the transpose). By doing so, the overall complexity of the algorithm is further reduced to $\mathcal{O}(mn\ln(n))$.

In our case, two combinations of $\mathbf{S}_1$ and $\mathbf{S}_2$ will be used to apply Algorithm 2 for the leverage score sampling, as summarised below in Table 3.2.1, where SRFT is the subsampled fast trigonometric transform (3.2.3) with $\mathbf{F}$ being the discrete cosine transform (DCT), JLT is the Johnson-Lindenstrauss transform (3.1.4) and Hashing is for the s-Hashing embeddings in (3.2.2) with $s = \max\{\lceil \ln(n) \rceil, 2\}$.

**Table 3.2.1** Two combinations of parameters in leverage score approximation

| Type | $\mathbf{S}_1$ | $d_1$ | $\mathbf{S}_2^{\mathrm{T}}$ | $d_2$ |
|---|---|---|---|---|
| ALev1 | SRFT | $4n$ | JLT | $\lceil 8\ln(m) \rceil$ |
| ALev2 | Hashing | $\lceil n\ln(n) \rceil$ | JLT | $\lceil 8\ln(m) \rceil$ |

From now on, we will refer to the leverage score sampling with estimated leverage scores calculated through Algorithm 2 using a combination of parameters of type 1 and type 2 as type 1 and type 2 approximate leverage score sampling to indicate the difference between the two methods. Leverage score sampling with exact leverage scores will be called exact leverage score sampling.

## 3.3 Block Sketching and the Sketching Error

In the context of LSFECM, a natural idea is whether sketching can be performed simultaneously while assembling the coefficient matrix **G** in Section 2.3.1 to further improve the efficiency of our (LSFECM + Sketch-and-Solve) framework. However, all the sketching techniques we introduced above are based on the assumption that the target matrix **A**, i.e., **G** in LSFECM, has already been fully assembled. Such incompatibility eventually leads us to consider the following approach of block sketching, which could be applied to the target matrix without knowing the complete information it contains. We propose our formulation for block sketches as follows.

**Definition 3.3.1. Block Sketching**
*Given an $m \times n$ real matrix $\mathbf{A} = [\ \underline{\mathbf{A}}_1\ ;\ \cdots ; \underline{\mathbf{A}}_{n_{wk}}\ ]$, such that $\underline{\mathbf{A}}_i \in \mathbb{R}^{m_i \times n}$ and $\sum_{i=1}^{n_{wk}} m_i = m$, we say the matrix $\mathbf{S} \in \mathbf{R}^{d \times m}$ is a $n_{wk}$ (left) block sketch of $\mathbf{A}$ if*

$$\mathbf{S} = \begin{bmatrix} \underline{\mathbf{S}}_1 & \mathbf{O} & \cdots & \mathbf{O} \\ \mathbf{O} & \underline{\mathbf{S}}_2 & \ddots & \vdots \\ \vdots & \ddots & \ddots & \mathbf{O} \\ \mathbf{O} & \cdots & \mathbf{O} & \underline{\mathbf{S}}_{n_{wk}} \end{bmatrix} \tag{3.3.1}$$

*for $\underline{\mathbf{S}}_i \in \mathbb{R}^{d_i \times m_i}$ and $\sum_{i=1}^{n_{wk}} d_i = d$. $\mathbf{O}$ are zero (sub)matrices of proper sizes. We call $n_{wk}$ the number of (co)workers and denote $\mathbf{S} := (\underline{\mathbf{S}}_1, \cdots, \underline{\mathbf{S}}_{n_{wk}})$. When $n_{wk} = 1$, the definition of 1 block sketch is equivalent to that of the whole sketching.*

The advantage of applying the above block sketching is straightforward: we can now decompose the whole sketching task into $n_{wk}$ sub-tasks and thus allow them to be integrated into our LSFECM assembly algorithm (Algorithm 1) to achieve parallel computation. The block-sketching method could also retain certain crucial structural properties of the original matrix, such as (almost) block-diagonality and sparsity; we demonstrate this in the following Fig.3.3.1. A similar comparison is also carried out for $\mathbf{G}_\Omega$ generated by the 1D and 2D LSFECMs which will be used later in Section 4.2 and 4.3, respectively; see Fig.C.2.1 and Fig.C.2.2 in Appendix C.2.

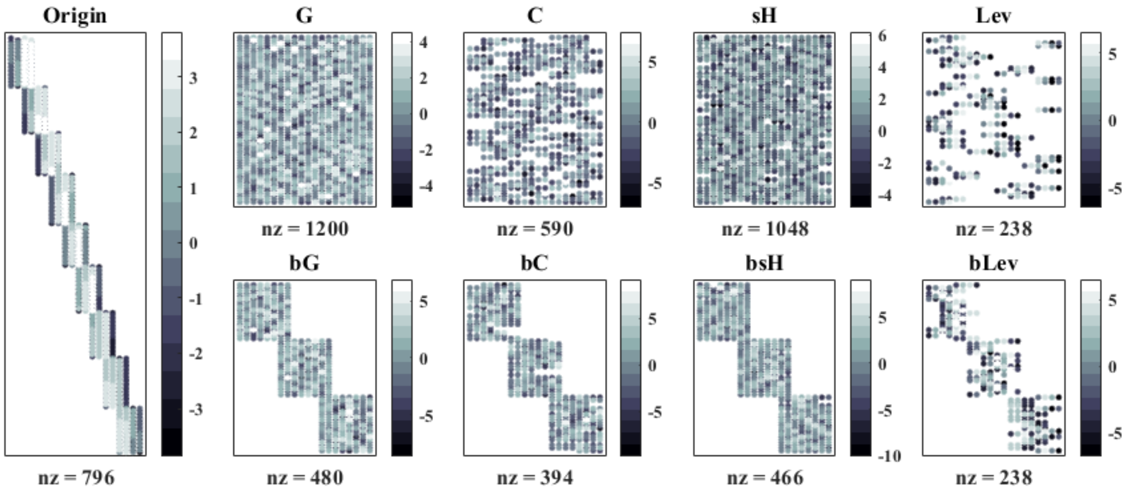

**Fig.3.3.1** Sketching on $200 \times 20$ almost block diagonal matrix (Origin). G: Gaussian projection; CS: Count Sketch; sH: Hashing embedding for $s = \lceil \ln(20) \rceil$; Lev: exact leverage score sampling; b- : corresponding block sketching with $n_{wk} = 3$.

To measure the quality of block sketching, the following theorem is proposed.

**Theorem 3.3.1. Quality of Block Sketching**
*Let* $\mathbf{S} = (\underline{\mathbf{S}}_1, \cdots, \underline{\mathbf{S}}_{n_{wk}})$ *be a* $n_{wk}$ *block sketch of some* $m \times n$ *matrix* $\mathbf{A} = [\ \underline{\mathbf{A}}_1\ ;\ \cdots ; \underline{\mathbf{A}}_{n_{wk}}\ ]$ *such that each* $\underline{\mathbf{S}}_i$ *admits an* $\epsilon_i$-SE *for* $\underline{\mathbf{A}}_i$. *Then,* $\mathbf{S}$ *is an* $\epsilon_{max}$-SE *for* $\mathbf{A}$ *where* $\epsilon_{max} = \max_{i \in [n_{wk}]} \epsilon_i$.

*Proof.* Let $\mathbf{y} = \mathbf{A}\mathbf{x} = [\ \underline{\mathbf{A}}_1\mathbf{x}\ ;\ \cdots ; \underline{\mathbf{A}}_{n_{wk}}\mathbf{x}\ ]$ for some $\mathbf{x} \in \mathbb{R}^n$ and denote $\mathbf{y}_i := \mathbf{A}_i\mathbf{x}$. We have

$$\|\mathbf{S}\mathbf{y}\|_2^2 = \left\| \begin{bmatrix} \underline{\mathbf{S}}_1 & \mathbf{O} & \cdots & \mathbf{O} \\ \mathbf{O} & \underline{\mathbf{S}}_2 & \ddots & \vdots \\ \vdots & \ddots & \ddots & \mathbf{O} \\ \mathbf{O} & \cdots & \mathbf{O} & \underline{\mathbf{S}}_{n_{wk}} \end{bmatrix} \begin{bmatrix} \mathbf{y}_1 \\ \vdots \\ \vdots \\ \mathbf{y}_{n_{wk}} \end{bmatrix} \right\|_2^2 = \sum_{i=1}^{n_{wk}} \|\underline{\mathbf{S}}_i \mathbf{y}_i\|_2^2. \tag{3.3.2}$$

Since each $\underline{\mathbf{S}}_i$ is an $\epsilon_i$-SE for $\underline{\mathbf{A}}_i$ and $\mathbf{y}_i \in \operatorname{span}(\underline{\mathbf{A}}_i)$,

$$\sum_{i=1}^{n_{wk}} \|\underline{\mathbf{S}}_i \mathbf{y}_i\|_2^2 \;\le\; \sum_{i=1}^{n_{wk}} (1+\epsilon_i)\|\mathbf{y}_i\|_2^2 \;\le\; (1+\epsilon_{max})\sum_{i=1}^{n_{wk}} \|\mathbf{y}_i\|_2^2 = (1+\epsilon_{max})\|\mathbf{y}\|_2^2, \quad (3.3.3)$$

$$\sum_{i=1}^{n_{wk}} \|\underline{\mathbf{S}}_i \mathbf{y}_i\|_2^2 \;\ge\; \sum_{i=1}^{n_{wk}} (1-\epsilon_i)\|\mathbf{y}_i\|_2^2 \;\ge\; (1-\epsilon_{max})\sum_{i=1}^{n_{wk}} \|\mathbf{y}_i\|_2^2 = (1-\epsilon_{max})\|\mathbf{y}\|_2^2. \quad (3.3.4)$$

Combining (3.3.2) – (3.3.4), we have shown that $\mathbf{S}$ provides an $\epsilon_{max}$-SE for $\mathbf{A}$. □

We further emphasise here that block sketching also preserves the two essential properties of sketching distributions [34, §2.2] in the following sense.

**Theorem 3.3.2.** *Let $\mathbf{S} = (\underline{\mathbf{S}}_1, \cdots, \underline{\mathbf{S}}_{n_{wk}})$ be a $n_{wk}$ block sketch such that $\underline{\mathbf{S}}_i \in \mathbb{R}^{d_i \times m_i}$ are drawn from distributions $\mathcal{S}_i$ where $\mathbb{E}[\underline{\mathbf{S}}_i] = \mathbf{O}$ (zero matrices) and $\mathbb{E}\|\underline{\mathbf{S}}_i \mathbf{y}_i\|_2^2 = \|\mathbf{y}_i\|_2^2$ for all $\mathbf{y}_i \in \mathbb{R}^{m_i}$. Then, for $m = \sum_{i=1}^{n_{wk}} m_i$ and arbitrary $\mathbf{y} \in \mathbb{R}^m$,*

$$\mathbb{E}[\mathbf{S}] = \mathbf{O} \quad \text{and} \quad \mathbb{E}\|\mathbf{S}\mathbf{y}\|_2^2 = \|\mathbf{y}\|_2^2. \quad (3.3.5)$$

The proof for the above Theorem 3.3.2 has been given in Appendix A.2 Theorem A.2.3. Finally, we introduce two error bounds for the sketch-and-solve method.

**Theorem 3.3.3. Error Bounds for Sketch-and-Solve Methods**
*Consider the linear least-squares problem $\mathbf{x}^* = \operatorname{argmin}_{\mathbf{x}\in\mathbb{R}^n} \|\mathbf{A}\mathbf{x} - \mathbf{b}\|_2^2$, where $\mathbf{A} \in \mathbb{R}^{m\times n}$ for $m \gg n$ is full rank. Let $\mathbf{S} \in \mathbb{R}^{d\times m}$ be an $\epsilon$-SE for $[\mathbf{A}, \mathbf{b}]$ and $\hat{\mathbf{x}} = \operatorname{argmin}_{\mathbf{x}\in\mathbb{R}^n} \|\mathbf{S}(\mathbf{A}\mathbf{x} - \mathbf{b})\|_2^2$, then the residual error $\mathrm{res}_{\mathbf{S}}$ satisfies*

$$\mathrm{res}_{\mathbf{S}} := \|\mathbf{A}\hat{\mathbf{x}} - \mathbf{b}\|_2 \;\le\; \sqrt{\frac{1+\epsilon}{1-\epsilon}}\|\mathbf{A}\mathbf{x}^* - \mathbf{b}\|_2, \quad (3.3.6)$$

*and the sketching error $\mathrm{err}_{\mathbf{S}}$ satisfies*

$$\mathrm{err}_{\mathbf{S}} := \|\hat{\mathbf{x}} - \mathbf{x}^*\|_2 \;\le\; \left(1 + \sqrt{\frac{1+\epsilon}{1-\epsilon}}\right) \|\mathbf{A}^\dagger\|_2 \cdot \|\mathbf{A}\mathbf{x}^* - \mathbf{b}\|_2. \quad (3.3.7)$$

*Note that $\|\mathbf{A}^\dagger\|_2$ is equal to the inverse of the smallest non-zero singular value of $\mathbf{A}$.*

*Proof.* For the first inequality (3.3.7), since $\hat{\mathbf{x}} = \operatorname{argmin}_{\mathbf{x}\in\mathbb{R}^n} \|\mathbf{S}(\mathbf{A}\mathbf{x} - \mathbf{b})\|_2^2$, we have

$$\|\mathbf{S}(\mathbf{A}\hat{\mathbf{x}} - \mathbf{b})\|_2^2 \;\le\; \|\mathbf{S}(\mathbf{A}\mathbf{x}^* - \mathbf{b})\|_2^2. \quad (3.3.8)$$

As $\mathbf{S}$ is an $\epsilon$-SE for $[\mathbf{A}, \mathbf{b}]$,

$$(1-\epsilon)\|\mathbf{A}\hat{\mathbf{x}} - \mathbf{b}\|_2^2 \;\le\; \|\mathbf{S}(\mathbf{A}\hat{\mathbf{x}} - \mathbf{b})\|_2^2, \quad (3.3.9)$$

$$(1+\epsilon)\|\mathbf{A}\mathbf{x}^* - \mathbf{b}\|_2^2 \;\ge\; \|\mathbf{S}(\mathbf{A}\mathbf{x}^* - \mathbf{b})\|_2^2. \quad (3.3.10)$$

Combining (3.3.8) – (3.3.10) gives (3.3.6). Then, for the second inequality (3.3.7), since $\mathbf{A}$ is full rank, there is

$$\begin{aligned} \|\hat{\mathbf{x}} - \mathbf{x}^*\|_2 &= \|\mathbf{A}^\dagger \mathbf{A}(\hat{\mathbf{x}} - \mathbf{x}^*)\|_2 \ \leq \ \|\mathbf{A}^\dagger\|_2 \cdot \|\mathbf{A}(\hat{\mathbf{x}} - \mathbf{x}^*)\|_2 \\ &= \|\mathbf{A}^\dagger\|_2 \cdot \|\mathbf{A}\hat{\mathbf{x}} - \mathbf{b} + \mathbf{b} - \mathbf{A}\mathbf{x}^*\|_2 \\ &\leq \|\mathbf{A}^\dagger\|_2 \cdot (\ \|\mathbf{A}\hat{\mathbf{x}} - \mathbf{b}\|_2 \ + \ \|\mathbf{A}\mathbf{x}^* - \mathbf{b}\|_2\ ). \end{aligned} \tag{3.3.11}$$

Combining (3.3.6) and (3.3.11) gives (3.3.7). □

## 3.4 Sketch-and-Solve in LSFECM

We close our discussion on this chapter by bringing the above sketch-and-solve method to our context of LSFECM. Recall that our formulation for the LSFECM in Section 2.3.1 would eventually lead us to a linear least-squares problem

$$\mathbf{c}^* = \underset{\mathbf{c} \in \mathbb{R}^{n_u n_v}}{\operatorname{argmin}} \|\mathbf{G}\mathbf{c} - \mathbf{d}\|_2^2 \tag{3.4.1}$$

where

$$\mathbf{G} = \begin{bmatrix} \mathbf{G}_\Omega \\ \mathbf{G}_\Gamma \end{bmatrix} \quad , \quad \mathbf{c} = \operatorname{vec}(\mathbf{C}) \quad , \quad \mathbf{d} = \begin{bmatrix} \mathbf{d}_\Omega \\ \mathbf{d}_\Gamma \end{bmatrix}. \tag{3.4.2}$$

*In order to retain the general structure of the augmented matrix* $[\mathbf{G}, \mathbf{d}]$*, and also to preserve the relative weights between the boundary and interior equations, we would sketch* $[\mathbf{G}_\Omega, \mathbf{d}_\Omega]$ *and* $[\mathbf{G}_\Gamma, \mathbf{d}_\Gamma]$ *separately.* Namely, the sketched problem is defined as finding $\hat{\mathbf{c}}$ such that

$$\hat{\mathbf{c}} = \underset{\mathbf{c} \in \mathbb{R}^{n_u n_v}}{\operatorname{argmin}} \left\| \begin{bmatrix} \mathbf{S}_\Omega & \mathbf{O} \\ \mathbf{O} & \mathbf{S}_\Gamma \end{bmatrix} \left( \begin{bmatrix} \mathbf{G}_\Omega \\ \mathbf{G}_\Gamma \end{bmatrix} \mathbf{c} - \begin{bmatrix} \mathbf{d}_\Omega \\ \mathbf{d}_\Gamma \end{bmatrix} \right) \right\|_2^2 \tag{3.4.3}$$

for the sketch

$$\mathbf{S} = \begin{bmatrix} \mathbf{S}_\Omega & \mathbf{O} \\ \mathbf{O} & \mathbf{S}_\Gamma \end{bmatrix} \quad \text{where} \quad \mathbf{S}_\Omega \in \mathbf{R}^{d_\Omega \times m_\Omega N_\Omega} \ , \ \mathbf{S}_\Gamma \in \mathbf{R}^{d_\Gamma \times m_\Gamma N_\Gamma}. \tag{3.4.4}$$

In some cases, when the interior equations are predominantly more than the boundary equations, i.e., when $m_\Omega N_\Omega \gg m_\Gamma N_\Gamma$, we would set $\mathbf{S}_\Gamma = \mathbf{I}_{m_\Gamma N_\Gamma}$ to be the identity matrix and only sketch the equations generated by the interior residuals. Using Theorem 3.3.1, $\mathbf{S} = (\mathbf{S}_\Omega, \mathbf{I}_{m_\Gamma N_\Gamma})$ provides an $\epsilon$-SE for $[\mathbf{G}, \mathbf{d}]$ as long as $\mathbf{S}_\Omega$ is an $\epsilon$-SE for $[\mathbf{G}_\Omega, \mathbf{d}_\Omega]$.

# Chapter 4

# Numerical Results for Sketch-and-Solve Methods

In this chapter, we apply the sketch-and-solve method introduced in Chapter 3 to solve discrete linear least-squares problems derived from the LSFECM in Chapter 2. We start our discussion by giving a short description on the basic experiment settings used in our implementation in Section 4.1.

## 4.1 Experiment Settings

The example problems we consider here are (2.3.22) – (2.3.23) for 1D and (2.3.32) – (2.3.36) for 2D, introduced in Sections 2.3.2 and 2.3.3, respectively. Collocation points are selected to ensure the corresponding least-squares problems (3.4.1) generated are sufficiently over-determined for sketching.

The sketching techniques used are Gaussian projections, Count Sketches, sHashing embeddings for oblivious subspace embeddings and (exact, type 1 and type 2 as described in Section 3.2.2) leverage score samplings for non-oblivious subspace embeddings. To distinguish the method of Count Sketch from the s-Hashing, the number of non-zeros in each column of the sketch for the latter method will always be set to be greater than 2, i.e., we set $s = \lceil \ln(n), 2 \rceil$ for every target matrix $\mathbf{A} \in \mathbb{R}^{m \times n}$.

All matrices with no more than 20% non-zero elements will be stored in sparse format, and all linear least-squares problems will be solved using the same QR algorithm, **solveLLS** in the implementation code, rather than directly applying the back-slash command in MATLAB to avoid extra disturbances caused by the built-in optimizations of the latter method. The running time we recorded for each method to compare is given in the following Table 4.1.1.

**Table 4.1.1** Running time recorded for comparison

| TIME | NO SKT | Whole SKT | | Block SKT | |
|---|---|---|---|---|---|
| | | OSE | NOSE | OSE | NOSE |
| Construction | – | – | √ | – | √ |
| Sketching | – | √ | √ | √ | √ |
| LLS Solving | √ | √ | √ | √ | √ |

Note that in the above table, we use "SKT" as a shorthand notation for "sketching". An approach that involves sketch-and-solve will be considered practical only if its total running time is shorter than that required to solve the original problem directly. We do not need to record the time for constructing the sketch for oblivious subspace embeddings (OSE), as we can always prepare them beforehand, while the case is not true for non-oblivious subspace embeddings (NOSE) as their construction depends on the data of the specific problem. *Note that in terms of block sketching, we will only record the time consumed by its most time-consuming subtask to emphasize the method's advantage of allowing parallel computation.*

We will use the notations $\mathbf{u}_h^* = \mathbf{C}^*\boldsymbol{\phi}$ and $\hat{\mathbf{u}}_h = \hat{\mathbf{C}}\boldsymbol{\phi}$ to denote the corresponding numerical solutions obtained by solving the complete and sketched least-squares problem (3.4.1) and (3.4.3), with $\mathbf{c}^* = \mathrm{vec}(\mathbf{C}^*)$ and $\hat{\mathbf{c}} = \mathrm{vec}(\hat{\mathbf{C}})$ unless otherwise specified. Finally, abbreviations will be used in all figures, with "G" for the Gaussian projection, "C" for the Count sketch, "sH" for the s-Hashing ($s \geq 2$), "Lev", "ALev1", "ALev2" for the exact, type 1 and type 2 (approximate) leverage score sampling, and "b" for the corresponding block sketching.

## 4.2 Application to 1D Example

For our 1D example, we construct a discrete linear least-squares problem (3.4.1) with a coefficient matrix $\mathbf{G}$ of size approximately equals to $10^5 \times 10^2$. We emphasize that, here in this example, we are selecting more collocation points than we actually needed solely for the convenience of the experiment. However, we also note that according to our parameter settings in Section 2.3.2, the row-column ratio of $\mathbf{G}$ will grow at an approximate rate of $\mathcal{O}(h^{-1})$, so in practice, a row-column ratio of 1000 is still achievable for $\mathbf{G}$ when the mesh is sufficiently fine. Here, we would follow the description in Section 3.4 and only sketch the interior equations $\mathbf{G}_\Omega$, i.e., set $\mathbf{S}_\Gamma$ as the identity for $\mathbf{S}$ in (3.4.4). Denoting the row-column ratio of $\mathbf{S}_\Omega\mathbf{G}_\Omega$ after sketching as $\hat{r}_\Omega$, the total runtime for our 1D experiments are recorded in Table 4.2.1 below. Note that in this example, we have $m_\Omega N_\Omega \gg m_\Gamma N_\Gamma = 2$ and thus readers free to estimate the row-column ratios of $\mathbf{G}$ and $\mathbf{SG}$ using the corresponding value of $\mathbf{G}_\Omega$ and $\mathbf{S}_\Omega\mathbf{G}_\Omega$.

**Table 4.2.1** Total running time ($\times 10^{-2}$ sec) for the 1D example

| SKT METHOD | $\hat{r}_\Omega$ | | | | | AVERAGE TIME |
|---|---|---|---|---|---|---|
| | 2 | 4 | 6 | 8 | 10 | |
| G | 5.89 | 12.5 | 19.1 | 25.9 | 32.5 | 19.18 |
| C | 0.59* | 0.77* | 0.92* | 1.12* | 1.42* | 0.96 |
| sH | 1.19* | 1.36* | 1.44* | 1.65* | 1.91* | 1.51 |
| Lev | 6.85 | 6.82 | 6.42 | 6.72 | 6.37 | 6.64 |
| ALev1 | 9.73 | 9.62 | 9.63 | 9.62 | 9.67 | 9.65 |
| ALev2 | 3.26 | 3.33 | 3.43 | 3.33 | 3.36 | 3.34 |
| 25 bG | 0.30* | 0.31* | 0.34* | 0.38* | 0.35* | 0.34 |
| 25 bC | 0.19* | 0.19* | 0.20* | 0.20* | 0.22* | 0.20 |
| 25 bSH | 0.21* | 0.22* | 0.21* | 0.22* | 0.22* | 0.22 |
| 25 bLev | 0.43* | 0.46* | 0.44* | 0.45* | 0.44* | 0.45 |
| 25 bALev1 | 1.27* | 1.26* | 1.22* | 1.24* | 1.22* | 1.24 |
| 25 bALev2 | 0.53* | 0.55* | 0.53* | 0.54* | 0.53* | 0.54 |

We utilize 25 coworkers to apply the block sketching. For each combination of the settings, 20 experiments are run to produce a median for the values recorded in the table. The median runtime for solving the exact least-squares problem (3.4.1) in this 1D example is $2.85 \times 10^{-2}$ sec. All sketch-and-solve strategies with a median runtime significantly less than that for solving (3.4.1) directly without sketching at a 1% significance level (in terms of the Wilcoxon rank sum test [24]) are highlighted with a "$*$" in the above table, from which we can see that all block sketching strategies the trivial choice of solving the complete least-squares problem. Yet, regarding the whole sketching, the only practical methods are the Count sketch and the s-Hashing embedding, while they are still much slower than their block-wise counterparts.

The reason why Gaussian projections is impractical is trivial: applying the dense Gaussian sketch $\mathbf{S} \in \mathbb{R}^{d\times m}$ already requires $\mathcal{O}(dmn)$ $(d > n)$ operations for any $m \times n$ target matrix $\mathbf{A}$ (where in our case is $\mathbf{G}_\Omega$); thus, using such a method does not actually reduce the complexity of solving the original least-squares problem $\min_{\mathbf{x}\in\mathbb{R}^n} \|\mathbf{A}\mathbf{x}-\mathbf{b}\|_2^2$.

For the whole-scale leverage score samplings, we find that the overall runtime does not vary with $\hat{r}_\Omega$. This is because the total runtime of this method is largely determined by that required to compute the leverage scores of the target matrix $\mathbf{A}$ ($\mathbf{G}_\Omega$ in our case). For the exact method, this is approximately equal to the time to compute the compact SVD of $\mathbf{A}$, which is $\mathcal{O}(mn^2)$; for the approximated methods, the time needed is further reduced to $\mathcal{O}(mn\ln(n))$, both independent of the sketching dimension $d$. The main reason why these methods fail to outperform the direct solver is that we use a "Q-less" QR [23] to solve sparse least-squares problems where the computation of complete QR decompositions is avoided, thus leading to a runtime significantly less than $\mathcal{O}(mn^2)$ for directly solving these sparse over-determined systems.

Readers may also notice that our type 1 approximate leverage score sampling here is not faster than the exact one. One possible explanation is that the problem itself is not large enough for the discrete cosine transform to outperform, as the same approximating algorithm soon becomes surpassing in our later experiments for 2D where the dimension of $\mathbf{G}_\Omega$ is further increased to $252000 \times 1101$. The explanation could also be used to interpret the same phenomenon in block sketching. Yet it is still worth mentioning that, even in the scenario described above, our type 2 approximate leverage score sampling, where Hashing matrices are used instead of the SRFTs in type 1 approximation, still reduced the runtime of exact leverage score samplings by nearly a half. The average misestimation factors given by the respective 100 samples of ALev1 and ALev2 are 0.62 and 0.63, respectively.

We also record the changes in percentage sparsities after the sketch-and-solve, i.e., psp($\mathbf{SG}$), compared to psp($\mathbf{G}$) in the following Table 4.2.2, where we define the percentage sparsity of a matrix $\mathbf{A} \in \mathbb{R}^{m\times n}$ as

$$\mathrm{psp}(\mathbf{A}) = \frac{\mathrm{nnz}(\mathbf{A})}{mn} \times 100\%. \tag{4.2.1}$$

Here $\mathrm{nnz}(\mathbf{A})$ is the number of nonzero elements in $\mathbf{A}$.

**Table 4.2.2** Percentage sparsity psp($\mathbf{SG}$) (×%) for the 1D example

| SKT METHOD | $\hat{r}_\Omega$ | | | | | AVERAGE |
|---|---|---|---|---|---|---|
| | 2 | 4 | 6 | 8 | 10 | |
| G | 99.03 | 99.51 | 99.67 | 99.76 | 99.80 | 99.56 |
| C | 99.03 | 99.49 | 99.44 | 98.88 | 97.67 | 98.90 |
| sH | 99.03 | 99.51 | 99.67 | 99.76 | 99.80 | 99.56 |
| Lev | 3.98 | 3.99 | 3.99 | 4.00 | 4.00 | 3.99 |
| ALev1 | 3.98 | 3.99 | 3.99 | 4.00 | 4.00 | 3.99 |
| ALev2 | 3.98 | 3.99 | 3.99 | 4.00 | 4.00 | 3.99 |
| 25 bG | 7.78 | 7.81 | 7.82 | 7.83 | 7.83 | 7.81 |
| 25 bC | 7.78 | 7.81 | 7.82 | 7.81 | 7.80 | 7.81 |
| 25 bsH | 7.78 | 7.81 | 7.82 | 7.83 | 7.83 | 7.81 |
| 25 bLev | 3.98 | 3.99 | 3.99 | 4.00 | 4.00 | 3.99 |
| 25 bALev1 | 3.98 | 3.99 | 3.99 | 4.00 | 4.00 | 3.99 |
| 25 bALev2 | 3.98 | 3.99 | 3.99 | 4.00 | 4.00 | 3.99 |

For each value displayed in the above table (except for the average), 20 experiments are carried out to provide a median. The percentage sparsity of the original coefficient matrix psp($\mathbf{G}$) is 4%. It can be observed that the scale of this value is well-preserved in all block sketching methods, as well as the leverage score samplings in terms of the whole sketching. However, as we have already seen in Table 4.2.1, the whole-scale leverage score samplings are not fast enough in our context.

We present changes in the sketching distortions $\epsilon$ estimated using Theorem 3.3.3

$$\epsilon \;\approx\; \hat{\epsilon} := \frac{\tau - 1}{\tau + 1} \quad \text{for} \quad \tau = \frac{\|\mathbf{G}\hat{\mathbf{c}} - \mathbf{d}\|_2^2}{\|\mathbf{G}\mathbf{c}^* - \mathbf{d}\|_2^2}, \tag{4.2.2}$$

as well as the sketching error $\|\hat{\mathbf{c}} - \mathbf{c}^*\|_2$, and the sketched $L^2$ error $\|\hat{\mathbf{u}}_h - \mathbf{u}\|_{0,\Omega}$ with respect to the sketched row-column ratio $\hat{r}_\Omega$ in Fig.4.2.1 and Fig.4.2.2 below respectively. Each box plot consists of 20 samples. For an explicit demonstration of the residual errors $\|\mathbf{G}\hat{\mathbf{c}} - \mathbf{d}\|_2$, We refer readers to Fig.C.2.3 in Appendix C.2.

In general, oblivious methods seem to outperform the non-oblivious methods both in terms of convergence and stability. All 1D experiments applying oblivious methods achieve an approximate distortion $\hat{\epsilon}$ within 0.5, while for those using non-oblivious methods, this is mostly achieved when $\hat{r}_\Omega \geq 6$. Among non-oblivious methods (both in whole sketching and block sketching), exact leverage score sampling provides the best performance in suppressing the growth of distortion, with ALev1 and ALev2 slightly less well-behaved. Yet, the differences among the effect of these (non-oblivious) methods soon become small when a sufficient number of rows ($\hat{r}_\Omega \geq 7$) are constructed in the sketch.

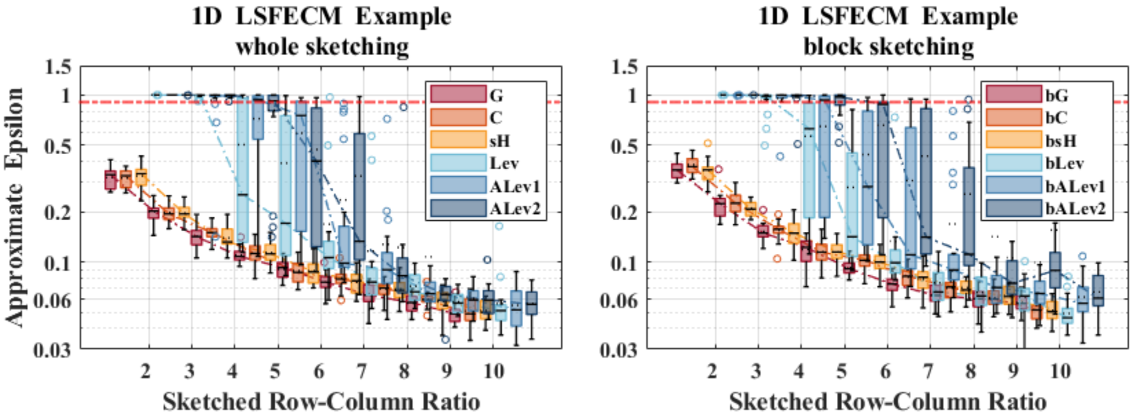

**Fig.4.2.1** (1D) Estimated sketching distortion $\epsilon$. Red dashed line: $\hat{\epsilon} = 0.9$.

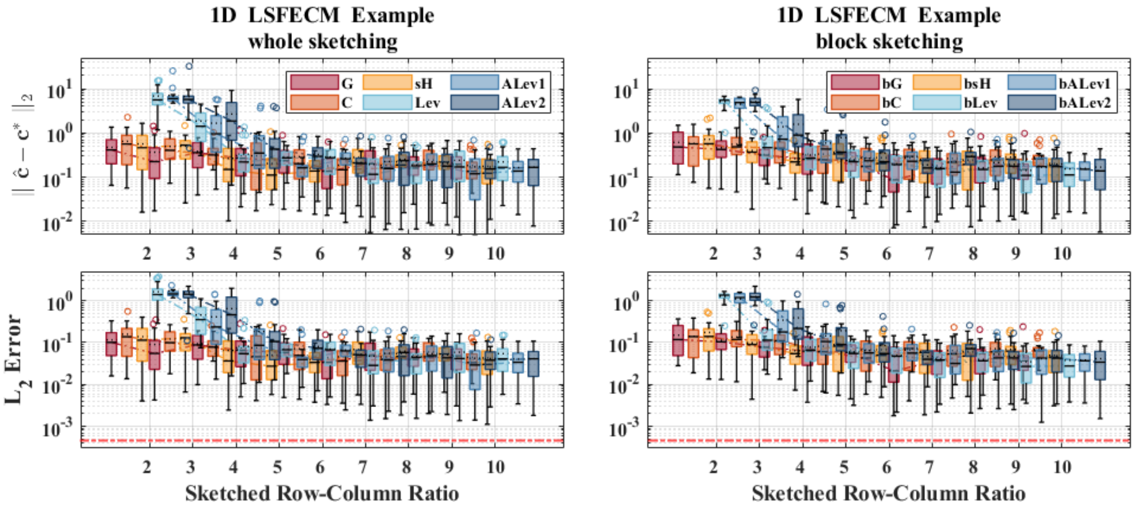

**Fig.4.2.2** (1D) Sketching error $\|\hat{\mathbf{c}} - \mathbf{c}^*\|_2$ displayed in the first row and sketched $L^2$ error $\|\hat{\mathbf{u}}_h - \mathbf{u}\|_{0,\Omega}$ displayed in the second row. Red dashed line: $\|\mathbf{u}_h^* - \mathbf{u}\|_{0,\Omega}$.

## 4.3 Application to 2D Example

In the 2D experiment, a discrete linear least-squares problem (3.4.1) was constructed using $\mathbf{G}$ with a size of $252216 \times 1101$ under the polygonal approximation of the domain (2.3.36), where $m_\Omega N_\Omega = 252000 \gg 216 = m_\Gamma N_\Gamma$. Again, more interior collocation points than needed, i.e., $\mathcal{O}(h^{-4})$, are generated to provide a sufficiently over-determined system with a relatively large $h$ for the convenience of the experiment.

**Table 4.3.1** Total running time ($\times 10^{-2}$ sec) for the 2D example

| SKT METHOD | $\hat{r}_\Omega$ | | | | | AVERAGE TIME |
|---|---|---|---|---|---|---|
| | 2 | 4 | 6 | 8 | 10 | |
| G | 335.2 | 730.9 | 1071. | 1522. | 2834. | 1299. |
| C | 39.53* | 126.7 | 34.40* | 46.12* | 55.30* | 60.41 |
| sH | 42.01* | 129.4 | 151.8 | 247.3 | 362.5 | 186.6 |
| Lev | 2259. | 2260. | 2260. | 2263. | 2264. | 2261. |
| ALev1 | 519.5 | 520.2 | 520.7 | 523.2 | 523.5 | 521.4 |
| ALev2 | 53.35* | 53.88* | 54.56* | 56.88* | 57.47* | 55.23 |
| 50 bG | 9.75* | 13.72* | 18.17* | 23.35* | 27.81* | 18.56 |
| 50 bC | 8.86* | 12.67* | 16.45* | 20.46* | 24.70* | 16.63 |
| 50 bsH | 8.96* | 12.90* | 17.12* | 21.43* | 25.97* | 17.28 |
| 50 bLev | 6.51* | 7.49* | 7.89* | 8.31* | 8.98* | 7.84 |
| 50 bALev1 | 6.94* | 7.74* | 8.38* | 8.65* | 9.29* | 8.20 |
| 50 bALev2 | 4.05* | 4.86* | 5.37* | 5.93* | 6.53* | 5.35 |

We again only sketch the interior equations as before and denote the row-column ratio after sketching as $\hat{r}_\Omega$. The total runtime for the 2D experiment is summarized in Table 4.3.1, where 50 coworkers are used to conduct the block sketching, and 20 experiments for each combination of the settings are run to produce a median for the values recorded. The median runtime for solving the complete least-squares problem is $81.43 \times 10^{-2}$ sec. Combinations of sketch-and-solve strategies with a faster speed at a 1% significance level regarding the Wilcoxon rank sum test are marked with a "$*$".

In this 2D scenario, we once again see that all block sketching strategies surpass the trivial strategy (without sketching) in terms of running speed. Nevertheless, results are slightly different this time regarding the whole sketching. For the oblivious methods, we see that most s-Hashing embeddings are no longer marked as practical. This is because the percentage sparsities after sketching using such a method have exceeded the threshold 0.2, which we set to apply the "Q-less" QR in solving the final least-squares problem. However, we also outline here that the percentage sparsity after sketching may be decreased by increasing the row numbers $d$ of the sketch $\mathbf{S} \in \mathbb{R}^{d\times m}$ when applied to sparse matrices. This explains the sudden drop in the runtime of the Count sketch method that occurred when $\hat{r}_\Omega$ increases from 4 to 6, which also results in a reduction in percentage sparsity (after sketching) from 0.25 to 0.18. For whole-scale leverage score samplings, we find that, at this time, both ALev1 and ALev2 admit significant speedups compared to the exact method, with ALev2's performance reaching a practical level.

We also record the percentage sparsities as we did in the previous section, where we observe that the scale of the value is once again well-preserved in all block sketching methods and the whole-scale leverage score samplings; see Table C.1.1 in Appendix C.1 for details. Changes in the estimated sketching distortion and the sketching error $\|\hat{\mathbf{c}} - \mathbf{c}^*\|_2$, together with the $L^2$ error $\|\hat{\mathbf{u}}_h - \mathbf{u}\|_{0,\Omega}$ with respect to $\hat{r}_\Omega$ are displayed in Fig.4.3.1 and Fig.4.3.2, with each box plot containing 20 samples. Plots on residual errors $\|\mathbf{G}\hat{\mathbf{c}} - \mathbf{d}\|_2$ are given by Fig.C.2.4 in Appendix C.2.

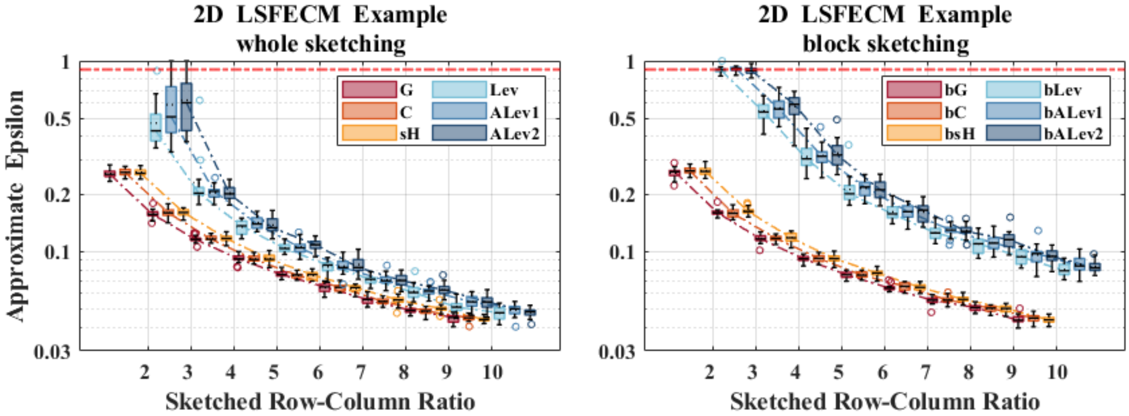

**Fig.4.3.1** (2D) Estimated sketching distortion $\epsilon$. Red dashed line: $\hat{\epsilon} = 0.9$.

Considering both experiments in 1D and 2D, we find that the effect of sketching is particularly stable regarding oblivious subspace embeddings. Whether these embeddings are applied whole-scale or block-wise, in 1D or 2D, the (approximated) sketching distortions generated remain below 0.5 even when the sketched row-column ratio is only 2. Further, whole-scale and block-wise methods also show relatively small dif-

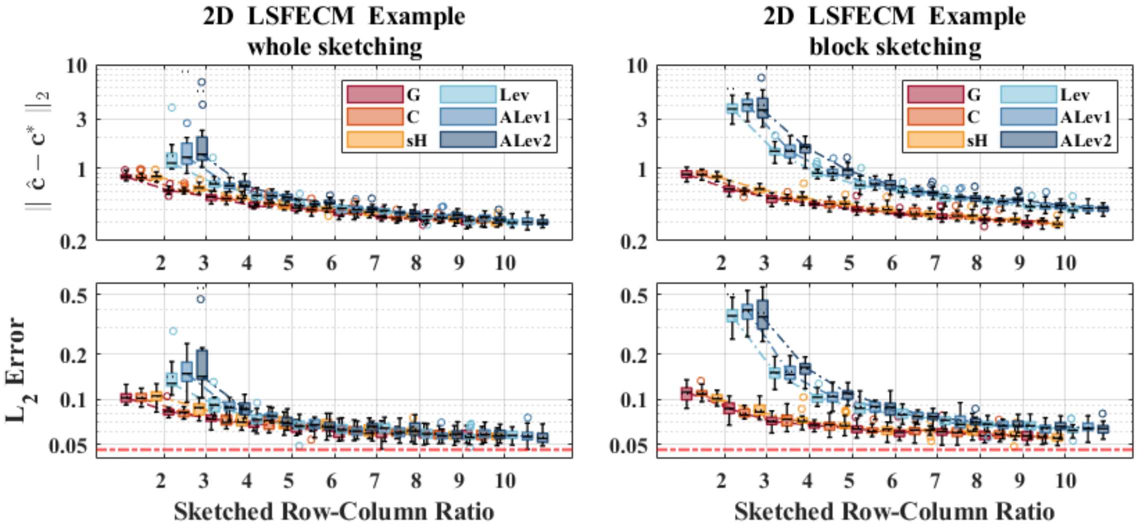

**Fig.4.3.2** Sketching error $\|\hat{\mathbf{c}}-\mathbf{c}^*\|_2$ displayed in the first row and sketched $L^2$ error $\|\hat{\mathbf{u}}_h-\mathbf{u}\|_{0,\Omega}$ displayed in the second row. Red dashed line: $\|\mathbf{u}_h^*-\mathbf{u}\|_{0,\Omega}$.

ferences in sketching quality in terms of oblivious methods.

However, the impact of transitioning from whole-scale sketches to corresponding block sketches is more pronounced in non-oblivious methods. In particular, in the 2D experiments, the convergence of distortion for block-wise non-oblivious sketching is noticeably slower compared to that for whole-scale sketching. This is likely due to the fact that block-wise sketching may potentially impair the consideration of global information for these non-oblivious methods. One possible way to circumvent this shortcoming is to perform a secondary sampling based on the importance of each block sketch, thereby enhancing the method's understanding towards global information, which we shall not discuss too much here in detail considering the page limitations.

To summarize our 1D and 2D attempts to solve derived linear least-squares problems of LSFECM with the sketch-and-solve method, we conclude that block sketching may always be a method worth considering due to its ability to allow parallel computation and preserve algebraic structures for the assembled system without largely changing the general sketching quality compared to their whole-scale counterparts. For whole-scale sketches, apart from the oblivious Hashing embeddings (both Count sketch and the s-Hashing), our non-oblivious ALev2 method may also provide relatively fast and reliable sketches when a large number of collocation points are selected to construct the final over-determined system for the LSFECMs.

# Chapter 5

# The Convergence and Error Analysis of LSFECM

In Section 2.1.1, the least-squares principle results in two general types of formulations: the pure continuous formulation (2.1.9),

$$\mathcal{I}_h^c(\mathbf{c}) = \|R_\Omega(\mathbf{x};\mathbf{c})\|_{Y(\Omega)}^2 + \|R_\Gamma(\mathbf{x};\mathbf{c})\|_{Y(\Gamma)}^2,$$

as well as the pure discrete formulation (2.1.10), also known as the over-determined collocation

$$\mathcal{I}_h^d(\mathbf{c}) = \textstyle\sum_{i=1}^{N_\Omega} \|R_\Omega(\mathbf{z}_i;\mathbf{c})\|_2^2 + \sum_{j=1}^{N_\Gamma} \|\mathbf{\Lambda} R_\Gamma(\hat{\mathbf{z}}_j;\mathbf{c})\|_2^2.$$

The general theory favouring the convergence analysis of continuous formulations has long been established with the proposal of LSFEMs. Yet, the corresponding theoretical tool for the discrete version is usually less considered in the literature. In this chapter, we show the extent to which oversampling can ensure the optimal convergence of LSFECM compared to its continuous counterparts in first-order linear boundary value problems (2.3.1).

## 5.1 Convergence in Least-Squares Methods

We note that both continuous (2.1.9) and discrete formulations (2.1.10) can be further classified depending on the conditions which the trial solution may satisfy. When the trial solutions $\mathbf{u}_h$ are chosen to meet the boundary condition, the methods are referred to as the *interior methods* as the boundary residuals vanish from all formulations of the residual error. In contrast, when the trial solutions automatically satisfy the interior condition, the methods are then called the *boundary methods*. Nevertheless, it is usually inconvenient or even impossible to find trial spaces whose elements may allow us to admit either one of these two approaches, in which case, the *mixed methods* (2.1.9) and (2.1.10) must be applied to find the final solution.

The convergence of continuous methods, including interior continuous [6] [25], boundary continuous [33] [42], and mixed continuous methods [6] [25] [32], has been extensively studied in various literature. Typically, these methods have the potential to converge both uniformly and quasi-optimally in the energy norm, under the existence of relation (2.1.2); see [6], [18], and [25].

However, analyses for convergence of discrete collocation methods are relatively less complete and could vary according to different choices of bases and collocation points [10, §3, §4]. In the following sections of the chapter, we will show that under certain assumptions, our oversampling LSFECM introduced in Section 2.3.1 for first-order linear boundary value problems (2.3.1) has the potential to converge at an optimal rate with respect to the mesh size $h$ of the $\mathrm{CG}_1$ finite elements; see Theorem 5.2.2.

## 5.2 Idea of Proof for the Convergence Rate

Our proof was inspired by the work of Maierhofer and Huybrechs in [30, §3]. For simplicity, we assume the discussion is carried out under the following settings: consider the first-order linear boundary value problem with zero Dirichlet boundary condition

$$\mathcal{A}[\mathbf{u}] := \sum_{i=1}^{n_d} \mathbf{A}_i \frac{\partial \mathbf{u}}{\partial x_i} + \mathbf{A}_0 \mathbf{u} = \mathbf{f}(\mathbf{x}) \quad \text{in} \quad \Omega \quad ; \quad \mathbf{u} = 0 \quad \text{on} \quad \Gamma. \tag{5.2.1}$$

Also, assuming the problem is fully $H^1$-coercive (as defined in Section 2.1.1), namely satisfying the energy balance relation below (with "$\lesssim$" as defined in Section 1.3.3)

$$\|\mathbf{u}\|_{1,\Omega} \lesssim \|\mathcal{A}[\mathbf{u}]\|_{0,\Omega} \lesssim \|\mathbf{u}\|_{1,\Omega} \; ; \tag{5.2.2}$$

for all $\mathbf{u} \in \mathbf{H}_0^1(\Omega)$. Further, for $N_\Omega$ (not necessarily uniform) interior collocation points $\{\mathbf{z}_i\}_{i=1}^{N_\Omega}$ , we defined the discrete $L^2$ inner product over the domain $\Omega$ as

$$\langle \mathbf{u}, \mathbf{v} \rangle_{0,N_\Omega} := \frac{m(\Omega)}{N_\Omega} \sum_{i=1}^{N_\Omega} \mathbf{u}^{\mathrm{T}}(\mathbf{z}_i)\mathbf{v}(\mathbf{z}_i), \tag{5.2.3}$$

with an induced norm

$$|[\mathbf{u}]|_{0,N_\Omega}^2 := \langle \mathbf{u}, \mathbf{u} \rangle_{0,N_\Omega}, \tag{5.2.4}$$

where $m(\Omega)$ denotes some measure for the region $\Omega$. The proof of our Theorem 5.2.2 was based on the observation that the LSFECM given in Section 2.3.1 amounts to the following discrete variational problem.

**Lemma 5.2.1.** *Least-squares problem* (2.3.15) – (2.3.16) *derived from the LSFECM in Section 2.3.1 for the boundary value problem* (5.2.1) *is equivalent to finding* $\mathbf{u}_h \in X_h(\Omega)$ *such that*

$$\mathrm{b}_N(\mathbf{u}_h, \mathbf{v}_h) = F_N(\mathbf{v}_h) \quad \text{for all} \quad \mathbf{v}_h \in X_h(\Omega), \tag{5.2.5}$$

*where* $X_h(\Omega)$ *denotes the product* $\mathrm{CG}_1$ *finite element space with vanishing boundary degrees of freedom, and*

$$\mathrm{b}_N(\mathbf{u}_h, \mathbf{v}_h) := \langle \mathcal{A}[\mathbf{u}_h], \mathcal{A}[\mathbf{v}_h] \rangle_{0,N_\Omega} \quad \text{and} \quad F_N(\mathbf{v}_h) := \langle \mathbf{f}, \mathcal{A}[\mathbf{v}_h] \rangle_{0,N_\Omega}. \tag{5.2.6}$$

*Proof.* Recall for $\mathbf{v}_h = \mathbf{C}_\mathbf{v}\boldsymbol{\phi} \in X_h$ and interior collocation points $\{\mathbf{z}_i\}_{i=1}^{N_\Omega}$, we have, as in Section 2.3.1,

$$\mathcal{A}[\mathbf{v}_h(\mathbf{z}_i)] = \begin{bmatrix} \mathcal{A}\left(\phi_1(\mathbf{z}_i)\right) & \cdots & \mathcal{A}\left(\phi_{n_v}(\mathbf{z}_i)\right) \end{bmatrix} \mathbf{c}_\mathbf{v} \tag{5.2.7}$$

for $\mathbf{c}_\mathbf{v} = \mathrm{vec}(\mathbf{C}_\mathbf{v})$. Denoting $\begin{bmatrix} \mathcal{A}\left(\phi_1(\mathbf{z}_i)\right) & \cdots & \mathcal{A}\left(\phi_{n_v}(\mathbf{z}_i)\right) \end{bmatrix}$ as $\widetilde{\mathbf{G}}(\mathbf{z}_i)$, equation (5.2.5) is equivalent to

$$\frac{m(\Omega)}{N_\Omega} \sum_{i=1}^{N_\Omega} \mathbf{c}_\mathbf{u}^\mathrm{T} \widetilde{\mathbf{G}}^\mathrm{T}(\mathbf{z}_i) \widetilde{\mathbf{G}}(\mathbf{z}_i) \mathbf{c}_\mathbf{v} = \frac{m(\Omega)}{N_\Omega} \sum_{i=1}^{N_\Omega} \mathbf{f}^\mathrm{T}(\mathbf{z}_i) \widetilde{\mathbf{G}}(\mathbf{z}_i) \mathbf{c}_\mathbf{v} \ . \tag{5.2.8}$$

Dividing both sides by $m(\Omega)/N_\Omega$ and taking the transpose, we have

$$\mathbf{c}_\mathbf{v}^\mathrm{T} \sum_{i=1}^{N_\Omega} \widetilde{\mathbf{G}}^\mathrm{T}(\mathbf{z}_i) \widetilde{\mathbf{G}}(\mathbf{z}_i) \mathbf{c}_\mathbf{u} = \mathbf{c}_\mathbf{v}^\mathrm{T} \sum_{i=1}^{N_\Omega} \widetilde{\mathbf{G}}^\mathrm{T}(\mathbf{z}_i) \mathbf{f}(\mathbf{z}_i) \ . \tag{5.2.9}$$

We require the relation (5.2.9) to hold for all $\mathbf{c}_\mathbf{v} \in \mathbb{R}^{n_v n_u}$, indicating

$$\sum_{i=1}^{N_\Omega} \widetilde{\mathbf{G}}^\mathrm{T}(\mathbf{z}_i) \widetilde{\mathbf{G}}(\mathbf{z}_i) \mathbf{c}_\mathbf{u} = \sum_{i=1}^{N_\Omega} \widetilde{\mathbf{G}}^\mathrm{T}(\mathbf{z}_i) \mathbf{f}(\mathbf{z}_i) \ . \tag{5.2.10}$$

Writing $\mathbf{G} = [\widetilde{\mathbf{G}}(\mathbf{z}_1); \cdots; \widetilde{\mathbf{G}}(\mathbf{z}_{N_\Omega})]$, $\mathbf{d} = [\mathbf{f}(\mathbf{z}_1); \cdots; \mathbf{f}(\mathbf{z}_{N_\Omega})]$, we again obtain the normal equation $\mathbf{G}^\mathrm{T}\mathbf{G} = \mathbf{G}^\mathrm{T}\mathbf{d}$ as we did in the end of Section 2.3.1. □

The case is the same if we include the inhomogeneous boundary terms. For boundary collocation points $\{\hat{\mathbf{z}}_j\}_{0,\Gamma}$ define discrete $L^2$ inner product over the boundary $\Gamma$ as

$$\langle \mathbf{u}, \mathbf{v} \rangle_{0,N_\Gamma} := \frac{m(\Gamma)}{N_\Gamma} \sum_{i=1}^{N_\Gamma} \mathbf{u}^\mathrm{T}(\hat{\mathbf{z}}_j) \mathbf{v}(\hat{\mathbf{z}}_j), \tag{5.2.11}$$

with an induced norm

$$||[\mathbf{u}]|^2_{0,N_\Gamma} := \langle \mathbf{u}, \mathbf{u} \rangle_{0,N_\Gamma}. \tag{5.2.12}$$

**Lemma 5.2.2.** *Least-squares problem* (2.3.15) – (2.3.16) *derived from the LSFECM in Section 2.3.1 for the boundary value problem* (2.3.1) *with a weight matrix*

$$\mathbf{\Lambda} = \sqrt{\frac{\lambda^2 N_\Omega m(\Gamma)}{N_\Gamma m(\Omega)}} \mathbf{I}_{m_\Gamma} \tag{5.2.13}$$

*is equivalent to finding* $\mathbf{u}_h \in X_h(\Omega)$ *such that*

$$\mathrm{b}_N(\mathbf{u}_h, \mathbf{v}_h) = F_N(\mathbf{v}_h) \quad \text{for all} \quad \mathbf{v}_h \in X_h, \tag{5.2.14}$$

*where* $X_h(\Omega)$ *denotes the product* $\mathrm{CG}_1$ *finite element space, and*

$$\mathrm{b}_N(\mathbf{u}_h, \mathbf{v}_h) := \langle \mathcal{A}[\mathbf{u}_h], \mathcal{A}[\mathbf{v}_h] \rangle_{0,N_\Omega} + \lambda^2 \langle \mathcal{B}[\mathbf{u}_h], \mathcal{B}[\mathbf{v}_h] \rangle_{0,N_\Gamma} \tag{5.2.15}$$

$$F_N(\mathbf{v}_h) := \langle \mathbf{f}, \mathcal{A}[\mathbf{v}_h] \rangle_{0,N_\Omega} + \lambda^2 \langle \mathbf{g}, \mathcal{B}[\mathbf{v}_h] \rangle_{0,N_\Gamma}. \tag{5.2.16}$$

The proof follows the same routine as what we did in the one for Lemma 5.2.1, and we shall omit it here in the paper. Next, define the following integral approximation error $\epsilon_I$ for the discrete inner product

$$\epsilon_I = \sup_{\mathbf{u},\mathbf{v} \in \mathbf{L}^2(\Omega)} \left| \langle \mathbf{u}, \mathbf{v} \rangle_{0,N_\Omega} - (\mathbf{u}, \mathbf{v})_{0,\Omega} \right|, \tag{5.2.17}$$

and recall the well-known result below for the interpolation error of triangular finite elements given by Braess [8, §6] on page 79 Theorem 6.4.

**Theorem 5.2.1. Interpolation Error for Triangular Elements [8, §6]** *Suppose the domain* $\Omega \subset \mathbb{R}^n$ *is discretised with a shape-regular triangulation, where all triangles are equipped with a* $C^0$*-conforming element consisting of piece-wise polynomials of degree* $t-1$*. Then for any* $u \in H^t(\Omega)$ *and* $0 \leq m \leq t$*,*

$$\|u - \Pi_h u\|_{m,\Omega} \lesssim h^{t-m} |u|_{t,\Omega} \tag{5.2.18}$$

*where* $\Pi_h$ *denotes the global interpolation operator,* $h$ *is the mesh size and* $|\cdot|_{t,\Omega}$ *represents the Soblev semi-norm.*

In our case, this means if $\mathbf{u} \in \mathbf{H}^2(\Omega)$ and $t = 2$, applying the above theorem allows us to expect

$$||\mathbf{u} - \Pi_h \mathbf{u}||_{1,\Omega} \lesssim h|\mathbf{u}|_{2,\Omega} \sim O(h). \tag{5.2.19}$$

We therefore conclude the following convergence requirement for the LSFECM.

**Theorem 5.2.2. Convergence of LSFECM for (5.2.1)**
*Let* $\mathbf{u} \in \mathbf{H}_0^1(\Omega)$ *be the true solution to the boundary value problem* (5.2.1) *as defined by Definition 2.1.1, and suppose that* $\mathbf{u}$ *is also in* $\mathbf{H}^2(\Omega)$*. If the problem* (5.2.1) *satisfies* (5.2.2)*, then* $\mathbf{u}_h$ *given by the LSFECM in Section 2.3.1 can converge quasi-optimally to* $\mathbf{u}$ *at rate* $\mathcal{O}(h)$ *w.r.t norm* $\|\cdot\|_{1,\Omega}$*, when the domain* $\Omega$ *is discretised with a shape-regular triangulation, and the integral approximation error* $\epsilon_I \sim \mathcal{O}(h^2)$*.*

*Proof.* Let $\mathbf{u}_h$ be the solution obtained via the LSFECM and for any $\mathbf{v}_h \in X_h \subset \mathbf{H}_0^1(\Omega)$, we have the energy balance relation

$$\|\mathbf{u}_h - \mathbf{v}_h\|_{1,\Omega}^2 \;\lesssim\; \|\mathcal{A}[\mathbf{u}_h - \mathbf{v}_h]\|_{0,\Omega}^2. \tag{5.2.20}$$

Let $\mathrm{b}(\mathbf{u}, \mathbf{v}) := (\mathcal{A}[\mathbf{u}], \mathcal{A}[\mathbf{v}])_{0,\Omega}$ and $F(\mathbf{v}) := (\mathbf{f}, \mathcal{A}[\mathbf{v}])_{0,\Omega}$ be as defined in Theorem 2.1.1 with $X(\Omega) := \mathbf{H}_0^1(\Omega)$ and $Y(\Omega) := \mathbf{L}^2(\Omega)$. Note that for $\|\mathcal{A}[\mathbf{u}_h - \mathbf{v}_h]\|_{0,\Omega}^2$ and $\mathbf{u} \in \mathbf{H}_0^1(\Omega)$, we have

$$\begin{aligned} \|\mathcal{A}[\mathbf{u}_h - \mathbf{v}_h]\|_{0,\Omega}^2 \;&=\; \mathrm{b}(\mathbf{u}_h - \mathbf{u}, \mathbf{u}_h - \mathbf{v}_h) + \mathrm{b}(\mathbf{u} - \mathbf{v}_h, \mathbf{u}_h - \mathbf{v}_h) \\ =\;& \mathrm{b}(\mathbf{u}_h, \mathbf{u}_h - \mathbf{v}_h) - F(\mathbf{u}_h - \mathbf{v}_h) + \mathrm{b}(\mathbf{u} - \mathbf{v}_h, \mathbf{u}_h - \mathbf{v}_h) \\ \le\;& \mathrm{b}_N(\mathbf{u}_h, \mathbf{u}_h - \mathbf{v}_h) - F_N(\mathbf{u}_h - \mathbf{v}_h) + \mathrm{b}(\mathbf{u} - \mathbf{v}_h, \mathbf{u}_h - \mathbf{v}_h) + 2\epsilon_I\ . \end{aligned} \tag{5.2.21}$$

Applying Lemma 5.2.1, we get $\mathrm{b}_N(\mathbf{u}_h, \mathbf{u}_h - \mathbf{v}_h) - F_N(\mathbf{u}_h - \mathbf{v}_h) = 0$, and thus,

$$\begin{aligned} \|\mathbf{u}_h - \mathbf{v}_h\|_{1,\Omega}^2 \;&\lesssim\; |\mathrm{b}(\mathbf{u} - \mathbf{v}_h, \mathbf{u}_h - \mathbf{v}_h)| + 2\epsilon_I \\ &\lesssim\; \|\mathbf{u} - \mathbf{v}_h\|_{1,\Omega} \cdot \|\mathbf{u}_h - \mathbf{v}_h\|_{1,\Omega} + 2\epsilon_I, \end{aligned} \tag{5.2.22}$$

by the continuity of $\mathrm{b}(\cdot,\cdot)$. Inequality (5.2.22) indicates that for some constant $\gamma > 0$, we have $\|\mathbf{u}_h - \mathbf{v}_h\|_{1,\Omega}^2 - \gamma(\|\mathbf{u} - \mathbf{v}_h\|_{1,\Omega} \cdot \|\mathbf{u}_h - \mathbf{v}_h\|_{1,\Omega} + 2\epsilon_I) \le 0$. Let $a = \|\mathbf{u}_h - \mathbf{v}_h\|_{1,\Omega}$, $b = \|\mathbf{u} - \mathbf{v}_h\|_{1,\Omega}$ and $c = 2\epsilon_I$. Since for $a, b, c, \gamma > 0$, $a^2 - \gamma ab - \gamma c \le 0$ implies $a \;\le\; \frac{1}{2}(\gamma b + \sqrt{\gamma^2 b^2 + 4\gamma c}) \;\lesssim\; b + \sqrt{c}$ (Appendix A.2, Lemma A.2.1), we have

$$\|\mathbf{u}_h - \mathbf{v}_h\|_{1,\Omega} \;\lesssim\; \|\mathbf{u} - \mathbf{v}_h\|_{1,\Omega} + \sqrt{\epsilon_I} \tag{5.2.23}$$

Using triangular inequality, and taking $\mathbf{v}_h := \Pi_h \mathbf{u}$ in (5.2.24) and applying (5.2.19), we have

$$\begin{aligned} \|\mathbf{u} - \mathbf{u}_h\|_{1,\Omega} \;&\le\; \|\mathbf{u} - \Pi_h \mathbf{u}\|_{1,\Omega} + \|\Pi_h \mathbf{u} - \mathbf{u}_h\|_{1,\Omega} \\ &\lesssim\; 2\|\mathbf{u} - \Pi_h \mathbf{u}\|_{1,\Omega} + \sqrt{\epsilon_I} \\ &\lesssim\; h|\mathbf{u}|_{2,\Omega} + \sqrt{\epsilon_I} \;\sim\; \mathcal{O}(\max\{h, \sqrt{\epsilon_I}\}). \end{aligned} \tag{5.2.24}$$

Thus, if the integral approximation error $\epsilon_I$ is around a scale of $\mathcal{O}(h^2)$, we could expect the LSFECM to converge quasi-optimally in norm $\|\cdot\|_{1,\Omega}$ at rate $\mathcal{O}(h)$. □

## 5.3 Integral Approximation: Choice of Samplers

As we can see from the above analysis, it is of vital importance for practitioners to choose collocation points in a suitable way so that $\epsilon_I$ in (5.2.17) can reach the desirable accuracy. Although we may sometimes apply uniform samplers to generate collocation points in our examples, it is worth noting that this kind of sampler is quite impractical in a general sense; see the following result of convergence for LSFECM using uniform samplers.

**Theorem 5.3.1. Convergence of LSFECM with Uniform Samplers**
*Under the conditions specified by Theorem 5.2.2, $N_\Omega \sim \mathcal{O}(h^{-4})$ uniformly sampled collocation points are sufficient to ensure the $\mathcal{O}(h)$ convergence under norm $\|\cdot\|_{1,\Omega}$.*

The theorem is a simple result deduced from the Central Limit Theorem and Chebyshev's inequality.

*Proof.* Suppose we want to estimate the integral $I_\varphi = \int_\Omega \varphi(\mathbf{x})\mathrm{d}\mathbf{x}$ using point-wise evaluations. Let $m(\Omega)$ denote the (Lebesgue) measure of the domain and $\mathbb{I}_\Omega(\mathbf{x})$ represent the indicator function; if $\{\mathbb{X}_i\}_{i=1}^n$ are i.i.d uniform random variables with a probability density function $f_{\mathbb{X}}(\mathbf{x}) = \frac{1}{m(\Omega)} \cdot \mathbb{I}_\Omega(\mathbf{x})$, we have, by the Central Limit Theorem,

$$\overline{\varphi(\mathbb{X})} := \frac{1}{n}\sum_{i=1}^{n} \varphi(\mathbb{X}_i) \;\longrightarrow\; \mathbb{E}[\varphi(\mathbb{X})] = \frac{I_\varphi}{m(\Omega)} \tag{5.3.1}$$

almost surely as $n \to \infty$; also, the square error is given by

$$\mathbb{E}\left[\left(m(\Omega)\overline{\varphi(\mathbb{X})} - I_\varphi\right)^2\right] \;=\; \mathrm{Var}\left[m(\Omega)\overline{\varphi(\mathbb{X})} - I_\varphi\right] \;=\; \frac{m^2(\Omega)\cdot\sigma^2_{\varphi(\mathbb{X})}}{n}, \tag{5.3.2}$$

where $\sigma^2_{\varphi(\mathbb{X})}$ is the variance of $\varphi(\mathbb{X})$. Using Chebyshev's inequality

$$\mathbb{P}\left(\; \left|m(\Omega)\overline{\varphi(\mathbb{X})} - I_\varphi\right| \geq \epsilon_I \;\right) \;\leq\; \frac{m^2(\Omega)\cdot\sigma^2_{\varphi(\mathbb{X})}}{\epsilon_I^2 \cdot n}. \tag{5.3.3}$$

We wish the upper bound in (5.3.3) to be small enough such that the probability $p := \mathbb{P}(\; |m(\Omega)\overline{\varphi(\mathbb{X})} - I_\varphi| < \epsilon_I \;)$ is within a desirable scale close to 1. Assuming $m(\Omega)$, and $\sigma_{\varphi(\mathbb{X})} \sim \mathcal{O}(1)$ leads to

$$\epsilon_I \;\leq\; \frac{m(\Omega)\cdot\sigma_{\varphi(\mathbb{X})}}{\sqrt{n\cdot(1-p)}} \;\sim\; \mathcal{O}(n^{-1/2}(1-p)^{-1/2}). \tag{5.3.4}$$

Now replacing $\varphi(\mathbb{X})$ with $(\mathbf{u},\mathbf{v})_{0,\Omega}$ in (5.2.17), and $n$ with $N_\Omega$, respectively. Clearly, $N_\Omega \sim \mathcal{O}(h^{-4}/(1-p))$ is sufficient to give $\epsilon_I \sim \mathcal{O}(h^2)$ required in the context of Theorem 5.2.2. □

For 1D LSFECM, applying equal-spaced collocation points may be a more practical strategy, which is known to improve the convergence of integration to $\epsilon_I \sim \mathcal{O}(1/N_\Omega)$. By doing so, only $\mathcal{O}(h^{-2})$ collocation points are needed to achieve the desirable convergence. In 2D, a more efficient alternative to the uniform samplers can be provided by the Quasi-Monte Carlo methods [3].

## 5.4 Inhomogeneous Boundary Conditions

All the above analyses are carried out under the assumption of a null Dirichlet boundary condition. We now start to consider the case when the boundary conditions are given in a more general sense,

$$\mathcal{A}[\mathbf{u}(\mathbf{x})] := \sum_{i=1}^{n_d} \mathbf{A}_i \frac{\partial \mathbf{u}}{\partial x_i} + \mathbf{A}_0 \mathbf{u} = \mathbf{f}(\mathbf{x}) \quad \text{in} \quad \Omega \quad ; \quad \mathbf{B}_0 \mathbf{u} = \mathbf{g}(\mathbf{x}) \quad \text{in} \quad \Gamma. \tag{5.4.1}$$

Assuming that the above formulation (5.4.1) is given by an elliptic boundary value problem of Petrovsky type, and that the boundary condition satisfies the Shapiro-Lopatinskij condition, thus satisfying the energy balance relation [6, §4.2]

$$\|\mathbf{u}\|_{1,\Omega} \lesssim \|\textstyle\sum_{i=1}^{n} \mathbf{A}_i \mathbf{u}_{x_i} + \mathbf{A}_0 \mathbf{u}\|_{0,\Omega} + \|\mathbf{B}_0 \mathbf{u}\|_{1/2,\Gamma} \lesssim \|\mathbf{u}\|_{1,\Omega}. \tag{5.4.2}$$

The above relation leads to the minimisation of the norm-equivalence functional,

$$\mathcal{I}[\mathbf{u}] := \|\textstyle\sum_{i=1}^{n} \mathbf{A}_i \mathbf{u}_{x_i} + \mathbf{A}_0 \mathbf{u}_h - \mathbf{f}\|_{0,\Omega}^2 + \|\mathbf{B}_0 \mathbf{u} - \mathbf{g}\|_{1/2,\Gamma}^2. \tag{5.4.3}$$

For a practical purpose, in a finite-dimensional trial space $X_h(\Omega)$, the fractional norm in (5.4.3) is usually replaced by a mesh-dependent weighted $L^2$ norm [4], leading to a so-called "practical" version of (5.4.3)

$$\widetilde{\mathcal{I}}_h[\mathbf{u}_h] := \|\textstyle\sum_{i=1}^{n} \mathbf{A}_i (\mathbf{u}_h)_{x_i} + \mathbf{A}_0 \mathbf{u}_h - \mathbf{f}\|_{0,\Omega}^2 + h^{-1} \|\mathbf{B}_0 \mathbf{u}_h - \mathbf{g}\|_{0,\Gamma}^2, \tag{5.4.4}$$

which only involves practical $L^2$ norms. The replacement here is guaranteed by the inverse inequality [12, §3.2]. Apparently, the solution $\mathbf{u}_h$ obtained by minimising (5.4.4) may not necessarily be the one obtained via minimising (5.4.3) directly over $X_h(\Omega)$. However, focusing on the practical version could also lead to optimal convergence in $H^1$ norm [4]. The corresponding discrete version for the LSFECM is

$$\widetilde{\mathcal{I}}_h^d[\mathbf{u}_h] := |[\textstyle\sum_{i=1}^{n} \mathbf{A}_i (\mathbf{u}_h)_{x_i} + \mathbf{A}_0 \mathbf{u}_h - \mathbf{f}]|_{0,N_\Omega}^2 + h^{-1} \, |[\mathbf{B}_0 \mathbf{u}_h - \mathbf{g}]|_{0,N_\Gamma}^2 \, , \tag{5.4.5}$$

which amounts to minimizing

$$\mathcal{I}_h^d(\mathbf{c}) = \textstyle\sum_{i=1}^{N_\Omega} \|R_\Omega(\mathbf{z}_i; \mathbf{c})\|_2^2 + \sum_{j=1}^{N_\Gamma} \|\mathbf{\Lambda} R_\Gamma(\hat{\mathbf{z}}_j; \mathbf{c})\|_2^2 \tag{5.4.6}$$

for

$$\mathbf{\Lambda} = \sqrt{\frac{h^{-1}N_\Omega m(\Gamma)}{N_\Gamma m(\Omega)}}\mathbf{I}_{m_\Gamma} \;\sim\; \mathcal{O}(\sqrt{h^{-1}N_\Omega/N_\Gamma}) \cdot \mathbf{I}_{m_\Gamma}. \tag{5.4.7}$$

Note that the weighted method can also be applied to other types of first-order systems in addition to elliptic systems of Petrovsky type, including the 2D decomposed Possion system we considered in Section 2.3.3. Yet, we are unable to derive a similar convergence requirement for the integral approximation error here as we have done in Section 5.2. The key reason is that the "practical" least-squares functional (5.4.4) is constructed at the cost of giving up its original norm-equivalence nature in the complete trail space [4].

However, we can still, to some extent, regard the requirements proposed in Theorem 5.2.2 as a "rule of thumb" for achieving desired convergence, thus providing some guidance or insight for setting the proper parameters of the method. For instance, in 1D, we are choosing $N_\Omega \sim \mathcal{O}(h^{-2})$ equally spaced interior collocation points. Since the boundary residual can be calculated exactly using only two end nodes, the weight matrix in (5.4.6) should be

$$\mathbf{\Lambda} \;\sim\; \mathcal{O}(\sqrt{h^{-1}h^{-2}/2}) \cdot \mathbf{I}_{m_\Gamma} \;\sim\; \mathcal{O}(h^{-1.5}) \cdot \mathbf{I}_{m_\Gamma}. \tag{5.4.8}$$

Similarly, in 2D, we select $N_\Omega \sim \mathcal{O}(h^{-4})$ interior collocation points uniformly and also, $\mathcal{O}(h^{-2})$ equal-spaced nodes over each boundary. The weight matrix in (5.4.6) is therefore, again,

$$\mathbf{\Lambda} \;\sim\; \mathcal{O}(\sqrt{h^{-1}h^{-4}/h^{-2}}) \cdot \mathbf{I}_{m_\Gamma} \;\sim\; \mathcal{O}(h^{-1.5}) \cdot \mathbf{I}_{m_\Gamma}. \tag{5.4.9}$$

## 5.5 Effect of Sketching

In this section, we further discuss the potential impact that applying the sketch-and-slove method may have on the convergence of LSFECM. We summarise our findings in the following theorem.

**Theorem 5.5.1. Convergence of Sketched LSFECM**
*Consider the LSFECM under the setting of Section 2.3.1. Let* $\mathbf{u}_h^* = \mathbf{C}^*\boldsymbol{\phi}$ *and* $\hat{\mathbf{u}}_h = \hat{\mathbf{C}}\boldsymbol{\phi}$ *be the solutions obtained by solving* (3.4.1) *and* (3.4.3)*, respectively, with* $\mathbf{c}^* = \mathrm{vec}(\mathbf{C}^*)$*,* $\hat{\mathbf{c}} = \mathrm{vec}(\hat{\mathbf{C}})$*. If for the true solution* $\mathbf{u}$*,* $\|\mathbf{u}_h^* - \mathbf{u}\|_{m,\Omega} \;\sim\; \mathcal{O}(h^k)$ *for some* $m \in \{0,1\}$ *and* $k \in \mathbb{N}_0$*, then*

$$\|\hat{\mathbf{u}}_h - \mathbf{u}\|_{m,\Omega} \;\sim\; \mathcal{O}(\max\{h^k, \|\mathbf{c}^* - \hat{\mathbf{c}}\|_2\}). \tag{5.5.1}$$

*Proof.* Note that for CG$_1$ bases $\boldsymbol{\phi}$ over bounded polygonal domains $\Omega$, we have bounded norms $\|\boldsymbol{\phi}\|_{m,\Omega} \leq M$ for some $M > 0$ and $m \in \{0, 1\}$. Thus,

$$\begin{aligned}
\|\hat{\mathbf{u}}_h - \mathbf{u}\|_{m,\Omega} &\leq \|\hat{\mathbf{u}}_h - \mathbf{u}_h^*\|_{m,\Omega} + \|\mathbf{u}_h^* - \mathbf{u}\|_{m,\Omega} \\
&= \|(\hat{\mathbf{C}} - \mathbf{C}^*)\boldsymbol{\phi}\|_{m,\Omega} + \|\mathbf{u}_h^* - \mathbf{u}\|_{m,\Omega} \\
&\leq \|\hat{\mathbf{C}} - \mathbf{C}^*\|_2 \cdot \|\boldsymbol{\phi}\|_{m,\Omega} + \|\mathbf{u}_h^* - \mathbf{u}\|_{m,\Omega} \\
&\leq M \cdot \|\hat{\mathbf{C}} - \mathbf{C}^*\|_F + \|\mathbf{u}_h^* - \mathbf{u}\|_{m,\Omega} \lesssim \|\hat{\mathbf{c}} - \mathbf{c}^*\|_2 + h^k, \qquad (5.5.2)
\end{aligned}$$

which concludes the proof and $\|\hat{\mathbf{u}}_h - \mathbf{u}\|_{m,\Omega} \sim \mathcal{O}(\max\{h^k, \|\mathbf{c}^* - \hat{\mathbf{c}}\|_2\})$. □

The above Theorem 5.5.1 shows that the sketching error $\|\mathbf{c}^* - \hat{\mathbf{c}}\|_2$ needs to be controlled within the scale of $\mathcal{O}(h^k)$ to prevent impairing the overall convergence rate of the LSFECM.

## 5.6 Numerical Results for Convergence

In this subsection, we aim to illustrate the convergence of LSFECM under our parameter settings based on the above analyses, as well as to demonstrate the possible impact that sketching may have on the convergence rate. We will keep using the notations and abbreviations proposed in the final paragraph of Section 4.1 throughout our discussion here.

### 5.6.1 Convergence in 1D Implementation

Table 5.6.1 summarises the (rounded) parameters used for the 1D experiments on the example problem (2.3.22) – (2.3.23), where $r_\Omega$ is the rounded row-column ratio for $\mathbf{G}_\Omega$ before sketching. $N_\Omega = \lceil \max\{h^{-2}, 10\pi h^{-1}\} \rceil$, and $\boldsymbol{\Lambda} = h^{-1.5}$.

**Table 5.6.1** Parameters used in the 1D experiment

| $h$ | .0050 | .0075 | .010 | .025 | .050 | .075 | .100 |
|---|---|---|---|---|---|---|---|
| $N_\Omega \times 10^3$ | 40.8 | 18.4 | 10.4 | 1.76 | .694 | .463 | .342 |
| $N_\Gamma$ | 2 | 2 | 2 | 2 | 2 | 2 | 2 |
| $\boldsymbol{\Lambda} \times 10^3$ | 2.83 | 1.54 | 1.00 | .253 | .089 | .049 | .032 |
| $r_\Omega$ | 65 | 44 | 33 | 14 | 11 | 11 | 11 |

All discretisations are conducted using shape-regular triangulations (intervals). $N_\Omega$ is chosen to be at least $10\pi h^{-1}$ to ensure the system is sufficiently over-determined for sketching, and the interior collocation points are selected in an equally spaced manner.

As usual, we preserve the boundary equations $\mathbf{G}_\Gamma$ and only sketch the part $\mathbf{G}_\Omega$ corresponding to the interior residuals. As we can see from Table 5.6.1, the smallest row-column ratio for $\mathbf{G}_\Omega$ is only around 11, we will thus not apply Leverage-Score Samplings and their variants in our 1D experiment as the discrete systems generated here are too small for the methods to preserve the original rank. The row-column ratio $\hat{r}_\Omega$ for $\hat{\mathbf{G}}_\Omega$ after sketching, and the number of coworkers $n_{wk}$ for applying the block sketching are set to be $\hat{r}_\Omega = 4$ and $n_{wk} = 10$ throughout the experiment.

The numerical results are illustrated in Fig.5.6.1 in the log10 scale. The first row displays the convergence of the $H^1$ error $\|\mathbf{u}_h^* - \mathbf{u}\|_{1,\Omega}$ and $\|\hat{\mathbf{u}}_h - \mathbf{u}\|_{1,\Omega}$ with respect to the mesh size $h$, and the second row corresponds to the convergence of the $L^2$ error $\|\mathbf{u}_h^* - \mathbf{u}\|_{0,\Omega}$ and $\|\hat{\mathbf{u}}_h - \mathbf{u}\|_{0,\Omega}$. We also display the residual error and the coefficient sketching error in Fig.5.6.2 for completeness.

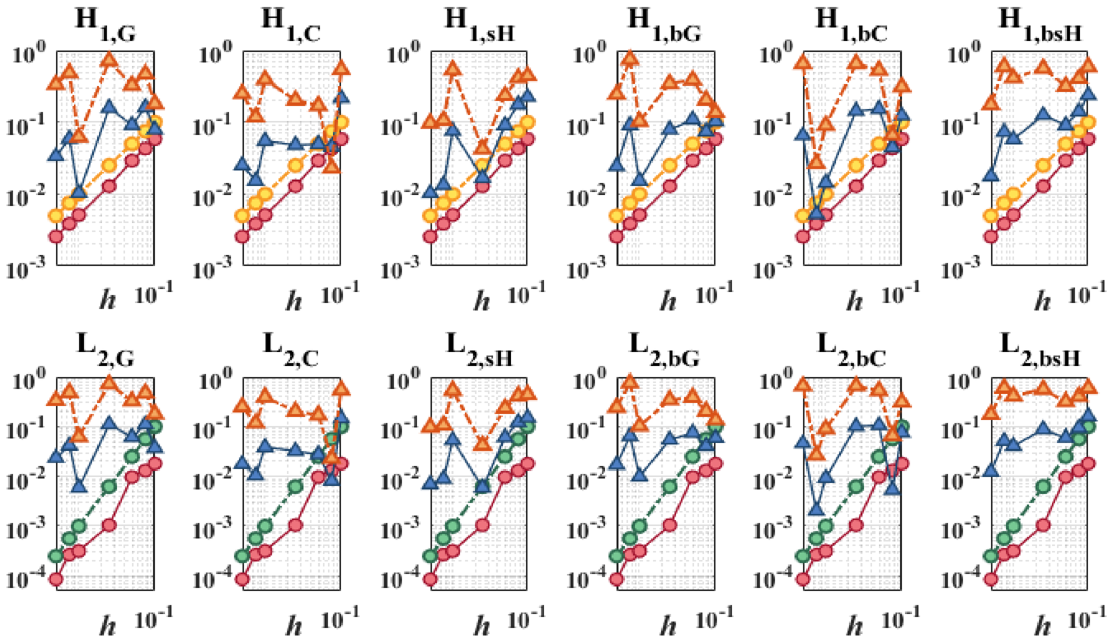

**Fig.5.6.1** Convergence of LSFECM applied to the 1D example problem. The first row displays the $H^1$ error $\|\mathbf{u}_h^* - \mathbf{u}\|_{1,\Omega}$ (red solid circles), $\|\hat{\mathbf{u}}_h - \mathbf{u}\|_{1,\Omega}$ (blue solid triangles) and $\mathcal{O}(h)$ (yellow dashed circles); the second row corresponds to the $L^2$ error $\|\mathbf{u}_h^* - \mathbf{u}\|_{0,\Omega}$ (red solid circles), $\|\hat{\mathbf{u}}_h - \mathbf{u}\|_{0,\Omega}$ (blue solid triangles) and $\mathcal{O}(h^2)$ (green dashed circles); orange dashed triangles: $\mathcal{O}(\|\hat{\mathbf{c}} - \mathbf{c}^*\|_2)$. $h$ labels are the same for each plot as given in Table 5.6.1, i.e., $h = [.005, .0075, .010, .025, .050, .075, .0100]$.

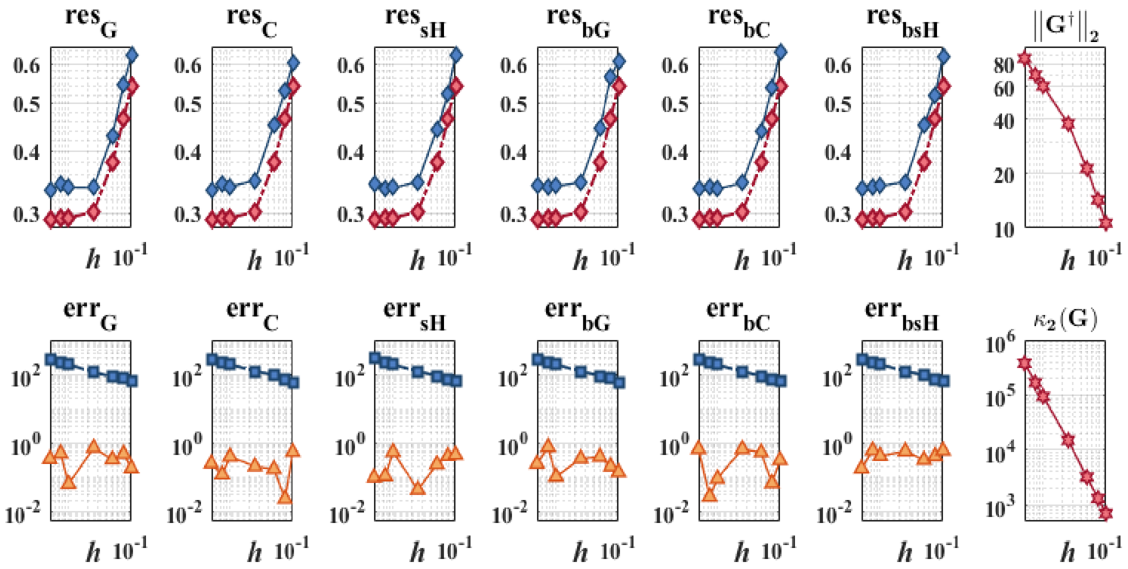

**Fig.5.6.2** Residuals after sketching $\|\mathbf{G}\hat{\mathbf{c}} - \mathbf{d}\|_2$ (blue solid diamonds), and coefficient sketching errors $\|\hat{\mathbf{c}} - \mathbf{c}^*\|_2$ (orange solid triangles) for the corresponding examples in Fig.5.6.1 displayed in log10 scale. Red dashed diamonds: residuals without sketching $\|\mathbf{G}\mathbf{c}^* - \mathbf{d}\|_2$; blue dashed squares: predicted bounds for coefficient errors. $h = [.005, .0075, .010, .025, .050, .075, .0100]$ are the same for all plots.

### 5.6.2 Convergence in 2D Implementation

Table 5.6.2 gives the (rounded) parameters used for the 2D experiments on the sample problem (2.3.32) – (2.3.36), with $N_\Omega = \lceil \max\{h^{-4}, 8 \cdot n_{ele}\} \rceil \sim \mathcal{O}(h^{-4})$, $N_\Gamma \sim \mathcal{O}(4h^{-2})$, and $\mathbf{\Lambda} = h^{-1.5}\mathbf{I}_2$. $n_{ele}$ is the number of elements in the mesh.

**Table 5.6.2** Parameters used in the 2D experiment

| $h$ | .050 | .075 | .100 | .125 | .150 | .175 | .200 |
|---|---|---|---|---|---|---|---|
| $N_\Omega \times 10^3$ | 164.2 | 33.01 | 10.65 | 6.896 | 5.040 | 3.680 | 2.903 |
| $N_\Gamma$ | 1594 | 711 | 396 | 266 | 197 | 154 | 121 |
| $\mathbf{\Lambda} \times \mathbf{I}_2$ | 89.4 | 48.7 | 31.6 | 22.6 | 17.2 | 13.7 | 11.2 |
| $r_\Omega$ | 78 | 34 | 19 | 19 | 18 | 18 | 17 |

The domain was discretised in a shape-regular manner. Again, for a mesh containing $n_{ele}$ elements, we set $N_\Omega = \lceil \max\{h^{-4}, 8 \cdot n_{ele}\} \rceil$ to ensure the system generated is sufficiently over-determined. Over each of the four boundaries, at least $h^{-2}$ equally spaced boundary collocation points will be selected. For sketching, only $\hat{\mathbf{G}}_\Omega$ will be sketched with a final row-column ratio $\hat{r}_\Omega = 4$. $n_{wk} = 50$ coworkers will be used to apply the block sketching. The results of 2D convergence experiments are demonstrated below in Fig.5.6.3 and Fig.5.6.4 below. Also see Appendix C.2, Fig.C.2.5 and Fig.C.2.6, for sketched residuals and coefficient errors.

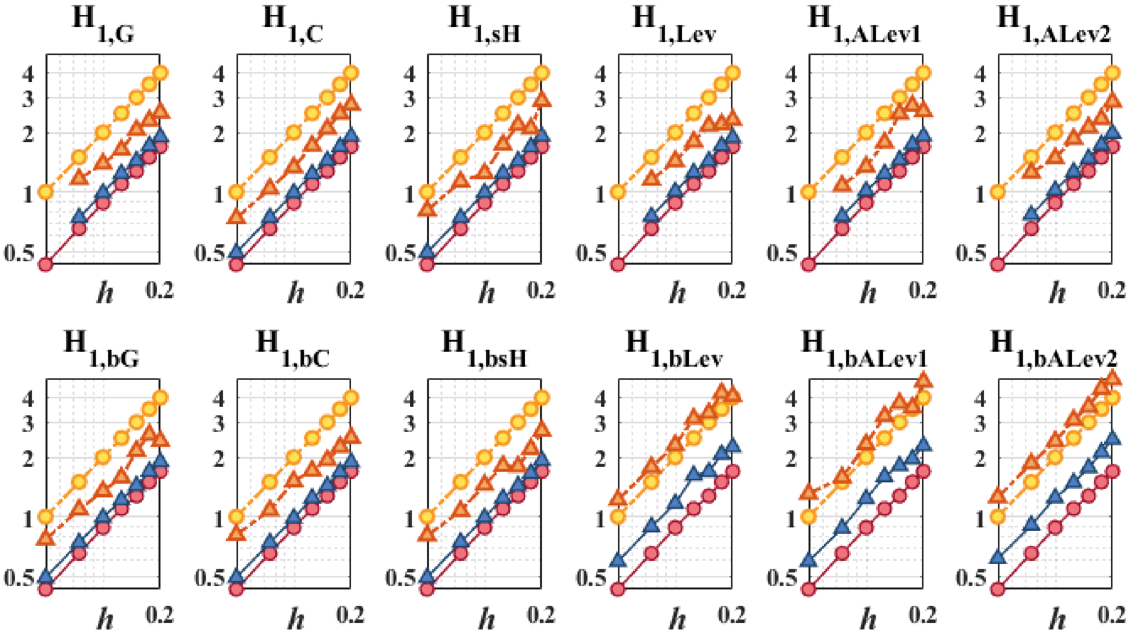

**Fig.5.6.3** Convergence of 2D LSFECM in $H^1$ error $\|\mathbf{u}_h^* - \mathbf{u}\|_{1,\Omega}$ (red solid circles), and $\|\hat{\mathbf{u}}_h - \mathbf{u}\|_{1,\Omega}$ (blue solid triangles), displayed in the log10 scale. Orange dashed triangles: $\mathcal{O}(\|\hat{\mathbf{c}} - \mathbf{c}^*\|_2)$; yellow dashed circles: $\mathcal{O}(h)$. $h$ labels are the same for each plot as given in Table 5.6.2, i.e., $h = [.050, .075, .100, .125, .150, .175, .200]$.

In terms of sketching, we noticed that some data from examples of Gaussian projections and leverage score samplings are missing, mainly because some of the steps in performing these methods exceeded the memory limit of the computer used for the experiment. For the Gaussian sketching, this happens because constructing and applying the dense Gaussian sketches is both memory- and time-consuming, while in the leverage score sampling, a more efficient way to perform the compact SVD for large sparse matrices, instead of MATLAB's function **svd**, which could only be used on full matrices, may be needed to circumvent the barrier. The aforementioned limitations of full-scale sketching further exemplify the advantages of block sketching in handling large-scale tasks by bringing in parallel computation, which could significantly enhance memory and time efficiency while only incurring limited accuracy loss.

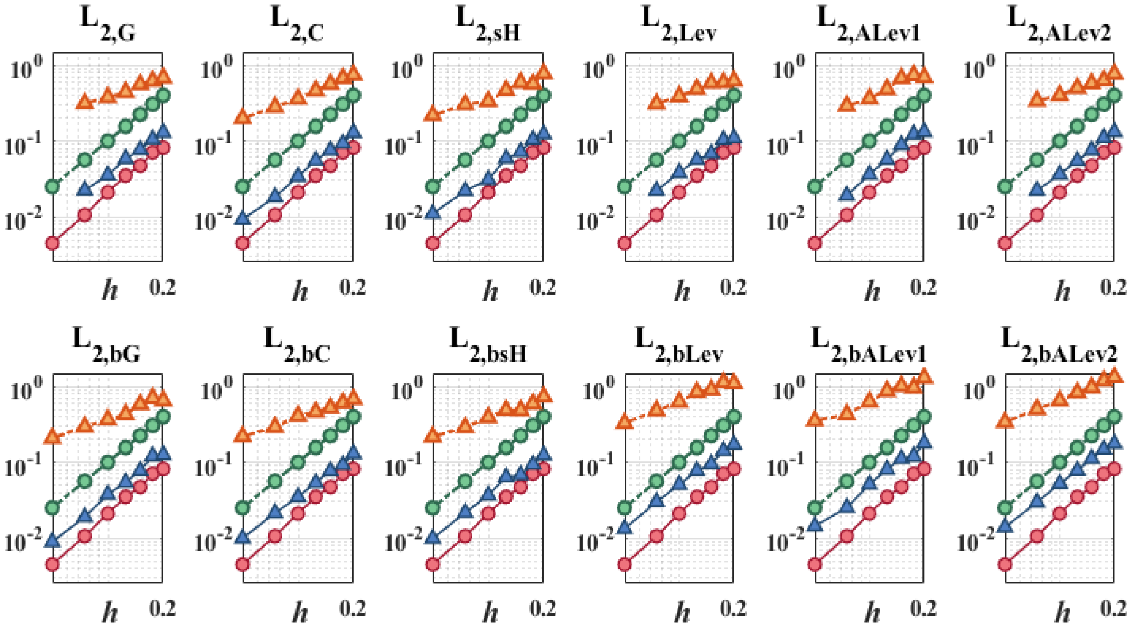

**Fig.5.6.4** Convergence of 2D LSFECM in $L^2$ error $\|\mathbf{u}_h^* - \mathbf{u}\|_{0,\Omega}$ (red solid circles), and $\|\hat{\mathbf{u}}_h - \mathbf{u}\|_{0,\Omega}$ (blue solid triangles), displayed in the log10 scale. Orange dashed triangles: $\mathcal{O}(\|\hat{\mathbf{c}} - \mathbf{c}^*\|_2)$; green dashed circles: $\mathcal{O}(h^2)$. $h$ labels are the same for each plot as given in Table 5.6.2, i.e., $h = [.050, .075, .100, .125, .150, .175, .200]$.

Regarding the convergence of LSFECM, the observations are mainly twofold. First, we see that under our own parameter settings, the model admits not only optimal convergence in $\|\cdot\|_{1,\Omega}$ but also near-optimal convergence in $\|\cdot\|_{0,\Omega}$ by solving the complete least-squares system (3.4.1) in both cases for 1D and 2D. Surprisingly, although the example problem in 2D is neither equipped with a homogeneous boundary condition nor defined on a simply connected region as we have required for being fully $H^1$-coercive, we still got optimal $\mathcal{O}(h^{-1})$ convergence in $\|\cdot\|_{1,\Omega}$, and $\mathcal{O}(h^{-2})$ convergence in $\|\cdot\|_{0,\Omega}$. We should also underline here that our expectation for observing optimal $\mathcal{O}(h)$ convergence for the above example problems is built on the fact that the true solutions are known to be in $\mathbf{H}^2(\Omega)$ by construction. Yet, this may not be true in the general sense for all first-order boundary value problems that one may encounter, and we thus leave it as an open problem.

Finally, in both 1D and 2D cases, the errors of the sketched LSFECM, $\|\hat{\mathbf{u}}_h - \mathbf{u}\|_{m,\Omega}$, are mostly bounded by the sketching errors of the basis coefficients $\mathcal{O}(\|\hat{\mathbf{c}} - \mathbf{c}^*\|_2)$, rather than the power of mesh sizes, as predicted by the Theorem 5.5.1, which implies that sufficient sketching is necessary to preserve the convergence rate of the method.

# Chapter 6

# Conclusion and Future Work

In this dissertation, we evaluate the potential of the (oversampling) LSFECM + Sketching framework in solving general first-order linear boundary value problems. We studied how sketching could be used in the context of LSFECM to provide a practical improvement and performed a detailed convergence analysis for the methods we proposed. There are four main conclusions we wish to emphasise here.

Firstly, we managed to make the sketching process compatible with parallel computation by introducing a technique called block sketching. Our numerical results in Chapter 4 showed that (sufficient) block sketching may significantly accelerate the solving process of the algebraic linear least-squares problem generated by the LSFECM without largely undermining the accuracy of the original solution.

Secondly, we improved the previously proposed algorithm for fast leverage score samplings [16, §3] by constructing FJLT with Hashing matrices instead of the SRFTs. We further experimentally demonstrated in Chapter 4 that such substitution could achieve evident speedup in the runtime compared to the original algorithm while also retaining the misestimation factor at a similar level.

Regarding the convergence analysis, a sufficient condition for the integral approximation error $\epsilon_I \sim \mathcal{O}(h^{-2})$ was derived for the oversampling LSFECMs to achieve quasi-optimal convergence for fully $H^1$-coercive first-order linear boundary value problems with zero Dirichlet boundary conditions. We further proved that such a condition could be achieved by selecting either $O(h^{-4})$ uniformly sampled collocation points (in 1D/2D) or $O(h^{-2})$ equally spaced collocation points in 1D. These estimations were then combined with the inverse inequality to provide a rule of thumb for the collocation weight matrix $\Lambda \sim \mathcal{O}(h^{-1.5}) \cdot \mathbf{I}_{m_\Gamma}$ for boundary value problems involving inhomogeneous boundary conditions.

Finally, we derived an overall error bound for the sketched LSFECMs, which is determined by both the original error bound (without sketching) $\mathcal{O}(h^k)$ as well as the sketching error for the bases coefficients $\mathcal{O}(\|\mathbf{c}^* - \hat{\mathbf{c}}\|_2)$, i.e., $\mathcal{O}(\max\{h^k, \|\mathbf{c}^* - \hat{\mathbf{c}}\|_2\})$. This bound, as well as the above coefficient estimations, together constitute the main theoretical contributions of this dissertation and were both verified via the two numerical experiments displayed in Section 5.6.

However, our work in this dissertation merely provides a foray into investigating the application of techniques in randomised linear algebra in solving differential equations, and there are still plenty of related topics that we could not delve into this time left for future research. For instance, according to the experiments carried out in Chapter 4, we have observed that whole-scale sketches generally failed to provide significant improvements in the runtime for solving linear least-squares problems derived from the LSFECM. This is partially due to the fact that these sketches may break the algebraic structure of the final coefficient matrix $\mathbf{G}$. Therefore, one extension to our work may be considering applying the sketch-and-solve to collocations where bases with global supports are used to construct the solution and lead to dense linear least-squares problems.

It may also be interesting if sketches could be used to construct preconditioners for some of the algebraic systems derived under the context of FEM or its variants, e.g., LSFEM and LSFECM etc. As we can see from the final plot in Fig.5.6.2, these (algebraic) systems may become growingly ill-conditioned when the meshes are continuously refined.

Lastly, in terms of the convergence of LSFECM, one possible theoretical improvement in future would be to extend Theorem 5.2.1 to the case of inhomogeneous boundary conditions. Another direction worth further research may be the attempt and development of more efficient sampling strategies for the choice of collocation points.

# Appendix A

# Notations and Proofs

## A.1 Notations

Below are some important notations which are not mentioned in Chapter 1 but are also used throughout this dissertation.

### A.1.1 Important Notations in Chapter 2

**1.** $n_d \in \{1, 2\}$ : dimension of the problem, i.e., number of dependent variables.

**2.** $n_u \in \mathbb{N}_+$ : number of independent variables.

**3.** $m_\Omega$, $m_\Gamma$ : number of interior and boundary equations, respectively.

**4.** $N_\Omega$, $N_\Gamma$ : number of interior and boundary collocation points, respectively.

**5.** $\mathbf{z}_i$, $\hat{\mathbf{z}}_j$ : interior and boundary collocation points, respectively.

**6.** $\mathbf{\Lambda}$ : diagonal matrix specifying the relative weights for each boundary residual.

**7.** $n_v$ : number of global vertices, or number of global degrees of freedom, or number of global bases, for the $\mathrm{CG}_1$ finite element space.

**8.** $X^c_{1,h}(\Omega)$ : continuous $\mathrm{CG}_1$ finite element spaces defined over $\Omega$ with mesh size $h$.

**9.** $X_h(\Omega)$ : discretised trial spaces. $X_h(\Omega) := [X^c_{1,h}(\Omega)]^{n_u}$ unless otherwise specified.

**10.** $\mathbf{u}_h(\mathbf{x}) \in X_h(\Omega)$ : the trial solution.

**11.** $\boldsymbol{\phi}(\mathbf{x}) \in \mathbb{R}^{n_v \times 1}$ : column vector of basis functions for CG1 finite element spaces.

12. $\mathbf{c}_i$, $\mathbf{C}$, $\mathbf{c}$ : $\mathbf{c}_i \in \mathbb{R}^{n_u \times 1}$ is the column vector of coefficients for the *ith* basis function in each component of $\mathbf{u}_h$, $\mathbf{C} = [\mathbf{c}_1, \cdots, \mathbf{c}_{n_v}] \in \mathbb{R}^{n_u \times n_v}$, and $\mathbf{c} = \mathrm{vec}(\mathbf{C}) \in \mathbb{R}^{n_u n_v \times 1}$.

13. $\mathbf{G}_\Omega$, $\mathbf{G}_\Gamma$ : the coefficient matrices derived from the interior and boundary residuals, respectively, for the final algebraic equation.

14. $\mathbf{d}_\Omega$, $\mathbf{d}_\Gamma$ : the output vectors derived from the interior and boundary residuals, respectively, for the final algebraic equation.

15. $\mathbf{G}$, $\mathbf{d}$, $\mathbf{c}^*$ : $\mathbf{G} = [\mathbf{G}_\Omega, \mathbf{G}_\Gamma]^{\mathrm{T}}$, $\mathbf{d} = [\mathbf{d}_\Omega, \mathbf{d}_\Gamma]^{\mathrm{T}}$, and $\mathbf{c}^*$ is the solution for the normal equation $\mathbf{G}^{\mathrm{T}}\mathbf{G}\mathbf{c}^* = \mathbf{G}^{\mathrm{T}}\mathbf{d}$.

### A.1.2 Important Notations in Chapter 3 – 6

1. $\mathbf{A}$, $m$, $n$ : $\mathbf{A} \in \mathbb{R}^{m \times n}$ is an arbitrary $m \times n$ real matrix where $m \gg n$.

2. $\mathbf{S}$, $d$ : $\mathbf{S} \in \mathbb{R}^{d \times m}$ is a (left) sketch for $\mathbf{A}$ with reduced row dimension $n < d \ll m$.

3. $\epsilon$ : subspace embedding distortion.

4. $\delta$ : highest probability for $(\epsilon, \delta)$-OSE or $(\epsilon, \delta)$-NOSE of not being a $\epsilon$-SE.

5. $s$ : number of nonzero entries in each column of the s-hashing matrix.

6. $\beta$ : misestimation factor for the approximate leverage score distribution.

7. $n_{wk}$ : number of (co)workers for block sketching.

8. $n_{ele}$ : number of elements in the $\mathrm{CG}_1$ mesh considered.

9. $\mathbf{S}_\Omega$, $\mathbf{S}_\Gamma$ : interior and boundary sketches applied to $\mathbf{G}_\Omega$ and $\mathbf{G}_\Gamma$ respectively.

10. $d_\Omega$, $d_\Gamma$ : row dimensions for $\mathbf{S}_\Omega$ and $\mathbf{S}_\Gamma$.

11. $r_\Omega$ : the row-column ratio of $\mathbf{G}_\Omega$, $r_\Omega = (m_\Omega N_\Omega)/(n_u n_v)$

12. $\hat{r}_\Omega$ : the row-column ratio of $\mathbf{S}_\Omega \mathbf{G}_\Omega$, $\hat{r}_\Omega = d_\Omega/(n_u n_v)$

13. $\hat{\mathbf{c}}$ : solution to the sketched least-squares problem $\underset{\mathbf{c} \in \mathbb{R}^{n_u n_v}}{\mathrm{argmin}} \|\mathbf{S}(\mathbf{G}\mathbf{c} - \mathbf{d})\|_2^2$.

14. $\mathbf{C}^*$, $\hat{\mathbf{C}}$, $\mathbf{u}_h^*$, $\hat{\mathbf{u}}_h$ : $\mathbf{c}^* = \mathrm{vec}(\mathbf{C}^*)$, $\hat{\mathbf{c}} = \mathrm{vec}(\hat{\mathbf{C}})$, $\mathbf{u}_h^* = \mathbf{C}^* \boldsymbol{\phi}$ and $\hat{\mathbf{u}}_h = \hat{\mathbf{C}} \boldsymbol{\phi}$.

15. $\epsilon_I$ : integral approximation error, $\epsilon_I = \sup_{\mathbf{u},\mathbf{v} \in \mathbf{L}^2(\Omega)} \left| \langle \mathbf{u}, \mathbf{v} \rangle_{0,N_\Omega} - (\mathbf{u}, \mathbf{v})_{0,\Omega} \right|$.

## A.2 Proofs

**Theorem A.2.1. Well-posedness of the LSFEM (Restate)** *Let* $\mathcal{A}[\cdot]$, $\mathcal{B}[\cdot]$, $\mathbf{f}$, $\mathbf{g}$, $\Omega$, $X(\Omega)$, $Y(\Omega)$, $Y(\Gamma)$ *and* $\mathcal{I}[\mathbf{u}]$ *be as defined in Definition 2.1.1. Then, minimizing* $\mathcal{I}[\mathbf{u}]$ *is equivalent to finding* $\mathbf{u} \in X(\Omega)$ *such that*

$$\mathrm{b}(\mathbf{u}, \mathbf{v}) = F(\mathbf{v}) \quad \text{for all} \quad \mathbf{v} \in X(\Omega), \tag{A.2.1}$$

*where*

$$\mathrm{b}(\mathbf{u}, \mathbf{v}) := (\mathcal{A}[\mathbf{u}], \mathcal{A}[\mathbf{v}])_{Y(\Omega)} + (\mathcal{B}[\mathbf{u}], \mathcal{B}[\mathbf{v}])_{Y(\Gamma)}, \tag{A.2.2}$$

$$F(\mathbf{v}) := (\mathbf{f}, \mathcal{A}[\mathbf{v}])_{Y(\Omega)} + (\mathbf{g}, \mathcal{B}[\mathbf{v}])_{Y(\Gamma)}. \tag{A.2.3}$$

*Furthermore, the bilinear form* $\mathrm{b}(\cdot, \cdot)$ *is both continuous and* $X(\Omega)$*-coercive; thus the problem* (A.2.1) – (A.2.3) *is well-posed by the Lax-Milgram theorem.*

*Proof.* Note the functional $\mathcal{I}[\mathbf{u}]$ is Gateaux differentiable and has the Gateaux derivative of

$$\mathcal{I}'[\mathbf{u}, \mathbf{v}] = (\mathcal{A}[\mathbf{u}] - \mathbf{f}, \mathcal{A}[\mathbf{v}])_{Y(\Omega)} + (\mathcal{B}[\mathbf{u}] - \mathbf{g}, \mathcal{B}[\mathbf{v}])_{Y(\Gamma)}, \tag{A.2.4}$$

for all $\mathbf{v} \in X(\Omega)$. A necessary condition to minimize $\mathcal{I}[\cdot]$ at $\mathbf{u}$ is obtained by forcing the Gateaux derivative to vanish at $\mathbf{u}$, thus leading us to (A.2.1) – (A.2.3).

The bilinear form $\mathrm{b}(\cdot, \cdot)$ is clearly symmetric, positive definite, and by the energy balance relation, is $X(\Omega)$-coercive

$$\begin{aligned} \mathrm{b}(\mathbf{u}, \mathbf{u}) &= \|\mathcal{A}[\mathbf{u}]\|^2_{Y(\Omega)} + \|\mathcal{B}[\mathbf{u}]\|^2_{Y(\Gamma)} \\ &\gtrsim \left(\|\mathcal{A}[\mathbf{u}]\|_{Y(\Omega)} + \|\mathcal{B}[\mathbf{u}]\|_{Y(\Gamma)}\right)^2 \\ &\gtrsim \|\mathbf{u}\|^2_{X(\Omega)}, \end{aligned} \tag{A.2.5}$$

and continuous

$$\begin{aligned} |\mathrm{b}(\mathbf{u}, \mathbf{v})| &= \left|(\mathcal{A}[\mathbf{u}], \mathcal{A}[\mathbf{v}])_{Y(\Omega)} + (\mathcal{B}[\mathbf{u}], \mathcal{B}[\mathbf{v}])_{Y(\Gamma)}\right| \\ &\leq \|\mathcal{A}[\mathbf{u}]\|_{Y(\Omega)} \cdot \|\mathcal{A}[\mathbf{v}]\|_{Y(\Omega)} + \|\mathcal{B}[\mathbf{u}]\|_{Y(\Gamma)} \cdot \|\mathcal{B}[\mathbf{v}]\|_{Y(\Gamma)} \\ &\leq \sqrt{\|\mathcal{A}[\mathbf{u}]\|^2_{Y(\Omega)} + \|\mathcal{B}[\mathbf{u}]\|^2_{Y(\Gamma)}} \cdot \sqrt{\|\mathcal{A}[\mathbf{v}]\|^2_{Y(\Omega)} + \|\mathcal{B}[\mathbf{v}]\|^2_{Y(\Gamma)}} \\ &\leq \left(\|\mathcal{A}[\mathbf{u}]\|_{Y(\Omega)} + \|\mathcal{B}[\mathbf{u}]\|_{Y(\Gamma)}\right) \cdot \left(\|\mathcal{A}[\mathbf{v}]\|_{Y(\Omega)} + \|\mathcal{B}[\mathbf{v}]\|_{Y(\Gamma)}\right) \\ &\lesssim \|\mathbf{u}\|_{X(\Omega)} \cdot \|\mathbf{v}\|_{X(\Omega)} \ . \end{aligned} \tag{A.2.6}$$

Similarly, one can prove that $F(\cdot) \in X^*(\Omega)$ is also continuous by the energy balance relation

$$\begin{aligned} |F(\mathbf{v})| &= \left|(\mathbf{f}, \mathcal{A}[\mathbf{v}])_{Y(\Omega)} + (\mathbf{g}, \mathcal{B}[\mathbf{v}])_{Y(\Gamma)}\right| \\ &\leq \|\mathbf{f}\|_{Y(\Omega)} \cdot \|\mathcal{A}[\mathbf{v}]\|_{Y(\Omega)} + \|\mathbf{g}\|_{Y(\Gamma)} \cdot \|\mathcal{B}[\mathbf{v}]\|_{Y(\Gamma)} \\ &\leq \sqrt{\|\mathbf{f}\|^2_{Y(\Omega)} + \|\mathbf{g}\|^2_{Y(\Gamma)}} \cdot \sqrt{\|\mathcal{A}[\mathbf{v}]\|^2_{Y(\Omega)} + \|\mathcal{B}[\mathbf{v}]\|^2_{Y(\Gamma)}} \\ &\lesssim \left(\|\mathcal{A}[\mathbf{v}]\|_{Y(\Omega)} + \|\mathcal{B}[\mathbf{v}]\|_{Y(\Gamma)}\right) \lesssim \|\mathbf{v}\|_{X(\Omega)} \,. \end{aligned} \tag{A.2.7}$$

According to the Lax-Milgram theorem, the variational formulation (A.2.1) – (A.2.3) (and thus, the minimization problem (2.1.3)) is well-posed. □

**Theorem A.2.2.** *For some $m \times n$ real matrix $\mathbf{A}$ with $\operatorname{rank}(\mathbf{A}) = r$, if $\mathbf{U} \in \mathbb{R}^{m\times r}$ satisfying $\operatorname{span}(\mathbf{U}) = \operatorname{span}(\mathbf{A})$ and $\mathbf{U}^{\mathrm{T}}\mathbf{U} = \mathbf{I}_r$, then the row leverage scores of $\mathbf{A}$ admit $\zeta_i = \|\mathbf{U}(i,:)\|_2^2$ for all $i \in [m]$.*

*Proof.* By Definition 3.2.4, we know $\operatorname{span}(\mathbf{U}) = \operatorname{span}(\mathbf{A}) = \operatorname{span}(\mathbf{U}_r)$. Thus, there exists some square matrix $\mathbf{R} \in \mathbb{R}^{r\times r}$ such that $\mathbf{U}_r = \mathbf{UR}$. Clearly,

$$\mathbf{U}_r^{\mathrm{T}}\mathbf{U}_r = \mathbf{R}^{\mathrm{T}}\mathbf{U}^{\mathrm{T}}\mathbf{U}\mathbf{R} = \mathbf{R}^{\mathrm{T}}\mathbf{R} = \mathbf{I}_r \;\Rightarrow\; \mathbf{R}^{\mathrm{T}}\mathbf{R} = \mathbf{I}_r. \tag{A.2.8}$$

Since $\mathbf{R}$ is square, $\mathbf{R}^{\mathrm{T}}\mathbf{R} = \mathbf{I}_r$ implies $\mathbf{R}\mathbf{R}^{\mathrm{T}} = \mathbf{I}_r$, indicating $\mathbf{R}$ is in fact orthogonal. Let $\mathbf{e}_i$ be the *ith* unit column vector; for arbitrary $i \in [m]$,

$$\begin{aligned} \zeta_i &= \|\mathbf{e}_i{}^{\mathrm{T}}\mathbf{U}_r\|_2^2 = \mathbf{e}_i{}^{\mathrm{T}}\mathbf{U}_r\mathbf{U}_r^{\mathrm{T}}\mathbf{e}_i = \mathbf{e}_i{}^{\mathrm{T}}\mathbf{U}\mathbf{R}\mathbf{R}^{\mathrm{T}}\mathbf{U}^{\mathrm{T}}\mathbf{e}_i \\ &= \mathbf{e}_i{}^{\mathrm{T}}\mathbf{U}\mathbf{U}^{\mathrm{T}}\mathbf{e}_i = \|\mathbf{e}_i{}^{\mathrm{T}}\mathbf{U}\|_2^2 = \|\mathbf{U}(i,:)\|_2^2, \end{aligned} \tag{A.2.9}$$

and we complete the proof for Theorem A.2.2. □

**Corollary A.2.1.** *The leverage score sampler $\mathbf{S} \in \mathbb{R}^{d\times m}$ in Definition 3.2.5 for the $m \times n$ real matrix $\mathbf{A}$ provides an $(\epsilon, \delta)$-NOSE if*

$$\beta d \geq [\ln(4/e)]^{-1}\epsilon^{-2} n \ln(2n/\delta). \tag{A.2.10}$$

*Proof.* It has already been proved in [34, §6.1] that $\mathbf{S}$ forms an $(\epsilon, \delta)$-NOSE if $\beta d \geq [(1+\epsilon)\ln(1+\epsilon) - \epsilon]^{-1} n \ln(2n/\delta)$. We only need to show that $\ln(4/e)\epsilon^2 \leq (1+\epsilon)\ln(1+\epsilon) - \epsilon$ for all $\epsilon \in [0,1]$. Let

$$f(\epsilon) = \ln(4/e)\epsilon^2 - (1+\epsilon)\ln(1+\epsilon) + \epsilon \quad \text{for} \quad \epsilon \in [0,1]. \tag{A.2.11}$$

We have

$$f'(\epsilon) = 2\ln(4/e)\epsilon - \ln(1+\epsilon), \tag{A.2.12}$$

$$f''(\epsilon) = 2\ln(4/e) - \frac{1}{1+\epsilon}. \tag{A.2.13}$$

It is easy to show that $f'(\epsilon)$ is strictly monotonically decreasing in $[0, \epsilon_*)$ and strictly monotonically increasing in $(\epsilon_*, 1]$ for $\epsilon_* = (2\ln(4/e))^{-1} - 1$. Since $f'(0) = 0$, $f'(1) = \ln(8/e^2) > 0$, $f'(\epsilon_*) < 0$ and there exist a zero $\epsilon_0 \in (\epsilon_*, 1)$ such that $f(\epsilon)$ is strictly monotonically decreasing in $[0, \epsilon_0)$ and strictly monotonically increasing in $(\epsilon_0, 1]$. Note that $f(0) = f(1) = 0$, and we therefore conclude $\ln(4/e)\epsilon^2 \leq (1+\epsilon)\ln(1+\epsilon) - \epsilon$ for all $\epsilon \in [0, 1]$, which finishes the proof for Corollary A.2.1. □

**Theorem A.2.3. (Restate)** *Let* $\mathbf{S} = (\underline{\mathbf{S}}_1, \cdots, \underline{\mathbf{S}}_{n_{wk}})$ *be a* $n_{wk}$ *block sketch such that* $\underline{\mathbf{S}}_i \in \mathbb{R}^{d_i \times m_i}$ *are drawn from distributions* $\mathcal{S}_i$ *where* $\mathbb{E}[\underline{\mathbf{S}}_i] = \mathbf{O}$ *(zero matrices) and* $\mathbb{E}\|\underline{\mathbf{S}}_i \mathbf{y}_i\|_2^2 = \|\mathbf{y}_i\|_2^2$ *for all* $\mathbf{y}_i \in \mathbb{R}^{m_i}$. *Then, for* $m = \sum_{i=1}^{n_{wk}} m_i$ *and arbitrary* $\mathbf{y} \in \mathbb{R}^m$,

$$\mathbb{E}[\mathbf{S}] = \mathbf{O} \quad \text{and} \quad \mathbb{E}\|\mathbf{S}\mathbf{y}\|_2^2 = \|\mathbf{y}\|_2^2. \tag{A.2.14}$$

*Proof.* The proof for $\mathbb{E}[\mathbf{S}] = \mathbf{O}$ is trivial, and we shall only show that $\mathbf{S}$ also preserves squared Euclidean norms in expectation. According to (3.3.2),

$$\mathbb{E}\|\mathbf{S}\mathbf{y}\|_2^2 \;=\; \mathbb{E}\left[\sum_{i=1}^{n_{wk}} \|\underline{\mathbf{S}}_i \mathbf{y}_i\|_2^2\right] \;=\; \sum_{i=1}^{n_{wk}} \mathbb{E}\|\underline{\mathbf{S}}_i \mathbf{y}_i\|_2^2 \;=\; \sum_{i=1}^{n_{wk}} \|\mathbf{y}_i\|_2 = \|\mathbf{y}\|_2^2. \tag{A.2.15}$$

□

**Lemma A.2.1.** *Given* $a, b, c, \gamma > 0$, $a^2 - \gamma ab - \gamma c \leq 0 \;\Rightarrow\; a \;\lesssim\; b + \sqrt{c}$.

*Proof.* Clearly, for $a, b, c, \gamma > 0$, $a^2 - \gamma ab - \gamma c \leq 0$ implies $a \;\leq\; \frac{1}{2}(\gamma b + \sqrt{\gamma^2 b^2 + 4\gamma c})$. Then we have,

$$\begin{aligned} a \;&\leq\; \frac{1}{2}(\gamma b + \sqrt{\gamma^2 b^2 + 4\gamma c}) \\ &\leq\; \frac{1}{2}\left(\gamma b + \sqrt{(\gamma b + 2\sqrt{\gamma c})^2}\right) \\ &=\; \gamma b + \sqrt{\gamma c} \;\lesssim\; b + \sqrt{c}, \end{aligned}$$

which completes the proof. □

# Appendix B

# Code and Implementation

## B.1 Code List

All codes for this programme can be found via this link. We provide some additional comments here for our main code implementation. We will mark with a “$*$” for those functions which may need further improvement for a more general application.

Before running the scripts below, please ensure the following packages or functions have been downloaded and added to the path: **parfor_progressbar_v1** (essential), **parTicToc** (essential), **200 colormap**, **special heatmap**, **daviolinplot** and **cspy**.

1. **Collocation1D** : Package to implement 1D LSFECM. Some key functions are :

   (1) **fitInterior** : fit interior equations;

   (2) **fitBoundary** : fit boundary equations;

   (3) **fitTrueSol** : fit true solutions if available;

   (4) **linear_discretize** : construct a simple uniform $CG_1$ mesh for the domain;

   (5) **findElement** : return the element id where the specific point is located;

   (6) **Assemble_B_Omega** : assemble $[\mathbf{G}_\Omega, \mathbf{d}_\Omega]$;

   (7) **Assemble_B_Gamma** : assemble $[\mathbf{G}_\Gamma, \mathbf{d}_\Gamma]$;

   (8) **evaluate_LSFEM** : evaluate $\mathbf{u}_h = \mathbf{C}\boldsymbol{\phi}$ according to the $\mathbf{C}$ given;

   (9) **Hk_error** : evaluate $\|\mathbf{u}_h - \mathbf{u}\|_{k,\Omega}$ for some $k \in \{0, 1\}$.

2. **Collocation2D** : Package to implement 2D LSFECM. This package retains all the key functions in **Collocation1D** as described above except for (4). Some additional features are :

   (1) **discretize*** : construct a simple uniform $\mathrm{CG}_1$ mesh for the domain;

   (2) **boundary_normal** : approximate outer normal vectors over the boundary;

   (3) **sample_from_triangleMesh** : sample interior collocation points uniformly;

   (4) **interp_polyboundary** : interpolate boundary collocation points into the boundary of the polygon.

3. **Sketchers** : Document of functions stored to apply sketching, including

   (1) **svdecon** : fast economical SVD written by Vijayan [38];

   (2) **FJLTransform** : construct a FJLT using discrete cosine transform;

   (3) **JLTransform** : construct a JLT using Theorem 3.1.1;

   (4) **leverageScore** : evaluate exact or fast approximation of leverage scores;

   (5) **sHashing** : construction of s-hashing matrices;

   (6) **leverageScoreSampling** : construction of leverage score samplers;

   (7) **blockEmbedding** : apply block sketches to the given matrix.

Some other functions and scripts are :

4. **solveLLS** : QR solver for linear least-squares problems;

5. **simplexquad** : calculate Gauss quadrature points and weights for simplices [40];

6. **Chapter2_1DLinearSys** : demonstration for the 1D example in Section 2.3.2;

7. **Chapter2_2DLinearSys** : demonstration for the 2D example in Section 2.3.3;

8. **Chapter4_SketchAndSolve** : script for numerical experiments in Chapter 4;

9. **Chapter5_Convergence_1D** : script for numerical experiments in Section 5.6.1;

10. **Chapter5_Convergence_2D** : script for numerical experiments in Section 5.6.2.

# Appendix C

# Tables and Images

## C.1 Tables

**Table C.1.1** Percentage sparsity psp(**SG**) (×%) for the 2D example in Section 4.3. Same value for the original coefficient matrix psp(**G**) is 0.54%.

| SKT METHOD | $\hat{r}_\Omega$ | | | | | AVERAGE |
|---|---|---|---|---|---|---|
| | 2 | 4 | 6 | 8 | 10 | |
| G | 91.08 | 95.33 | 96.84 | 97.61 | 98.08 | 95.79 |
| C | 40.97 | 25.07 | 17.95 | 13.95 | 11.41 | 21.87 |
| sH | 88.59 | 83.63 | 74.80 | 66.48 | 59.41 | 74.58 |
| Lev | 0.51 | 0.53 | 0.53 | 0.54 | 0.54 | 0.53 |
| ALev1 | 0.51 | 0.53 | 0.53 | 0.54 | 0.54 | 0.53 |
| ALev2 | 0.51 | 0.53 | 0.53 | 0.54 | 0.54 | 0.53 |
| 50 bG | 7.97 | 8.34 | 8.47 | 8.53 | 8.58 | 8.38 |
| 50 bC | 7.69 | 7.46 | 6.94 | 6.38 | 5.87 | 6.87 |
| 50 bsH | 7.96 | 8.31 | 8.41 | 8.44 | 8.43 | 8.31 |
| 50 bLev | 0.51 | 0.53 | 0.53 | 0.54 | 0.54 | 0.53 |
| 50 bALev1 | 0.51 | 0.53 | 0.53 | 0.54 | 0.54 | 0.53 |
| 50 bALev2 | 0.51 | 0.53 | 0.53 | 0.54 | 0.54 | 0.53 |

## C.2 Images

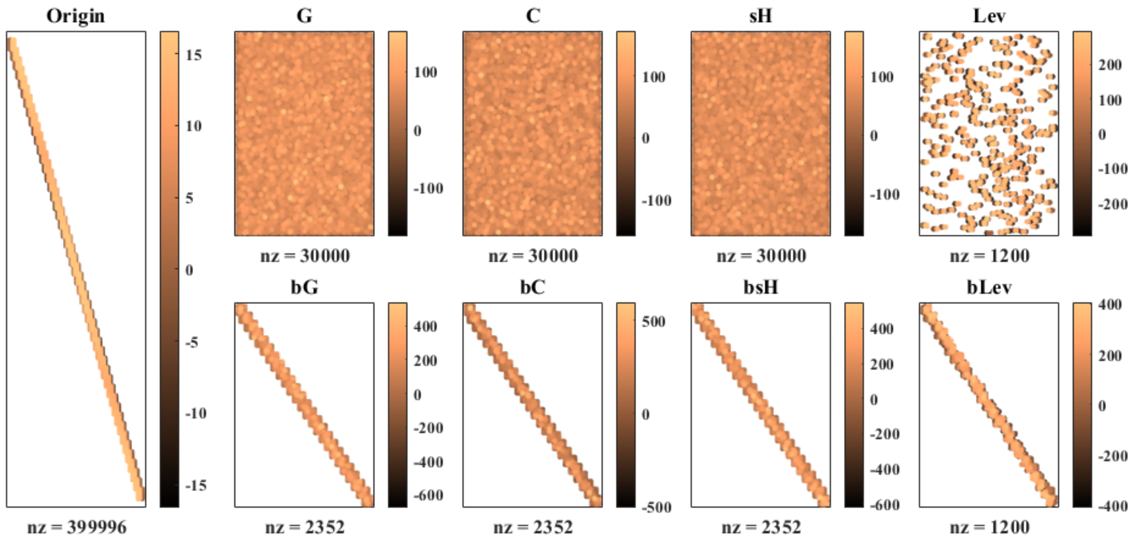

**Fig.C.2.1** Sketching on $10^5 \times 10^2$ $\mathbf{G}_\Omega$ (Origin) in Section 4.2. G: Gaussian projection; CS: Count Sketch; sH: Hashing embedding for $s = \lceil \ln(100) \rceil$; Lev: exact leverage score sampling; b- : corresponding block sketching with $n_{wk} = 25$.

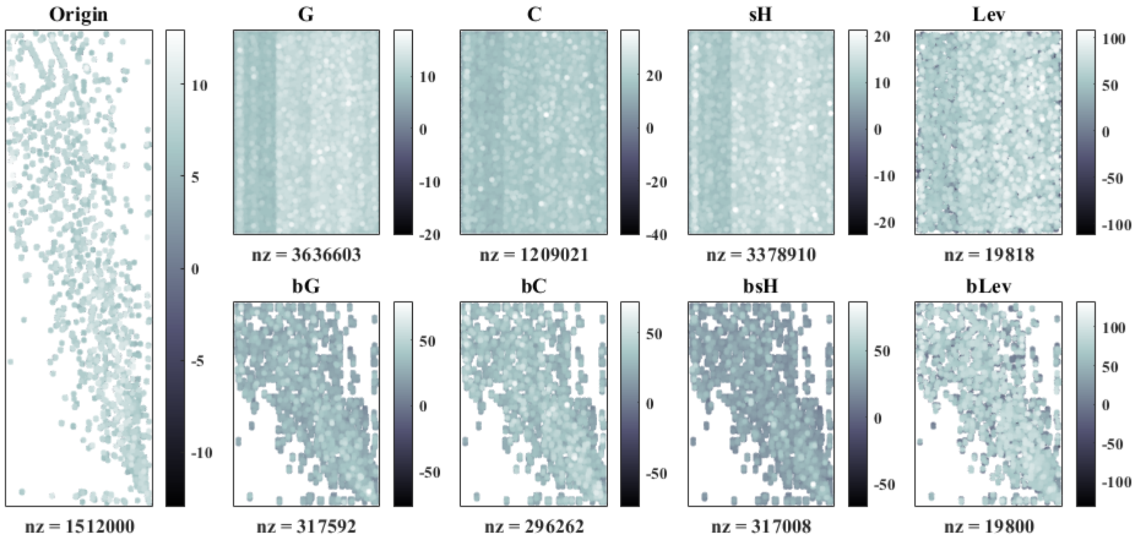

**Fig.C.2.2** Sketching on 252000 × 1101 $\mathbf{G}_\Omega$ (Origin) in Section 4.3. G: Gaussian projection; CS: Count Sketch; sH: Hashing embedding for $s = \lceil \ln(1101) \rceil$; Lev: exact leverage score sampling; b- : corresponding block sketching with $n_{wk} = 50$.

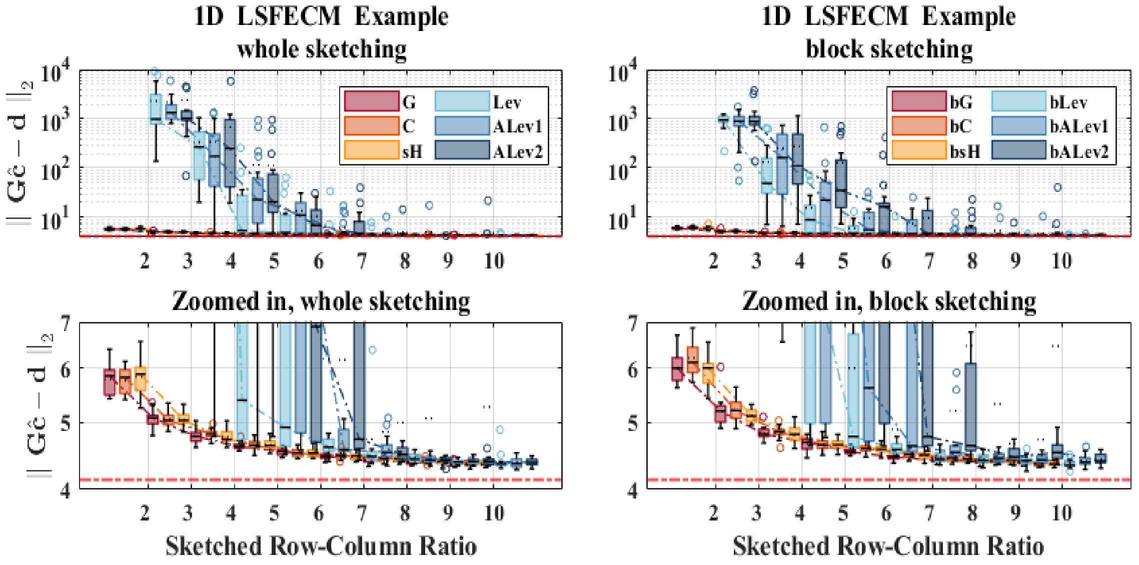

**Fig.C.2.3** Sketched residual error $\|\mathbf{G}\hat{\mathbf{c}} - \mathbf{d}\|_2$ for the 1D example in Section 4.2. Red dashed line: residual without sketching $\|\mathbf{G}\mathbf{c}^* - \mathbf{d}\|_2 \approx 4.14$; 20 samples for each box plot.

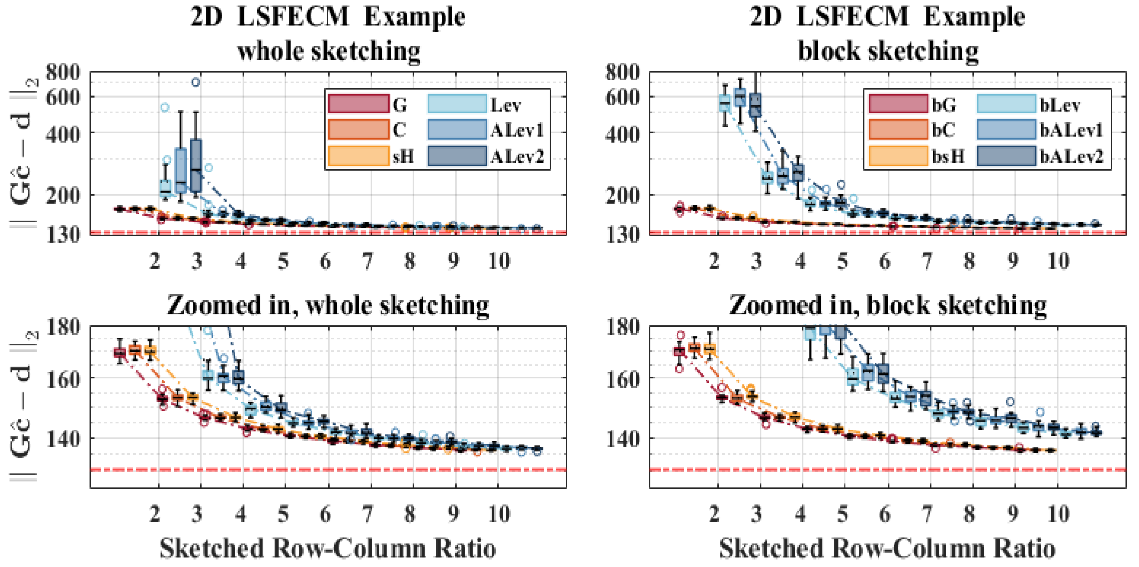

**Fig.C.2.4** Sketched residual error $\|\mathbf{G}\hat{\mathbf{c}} - \mathbf{d}\|_2$ for the 2D example in Section 4.3. Red dashed line: residual without sketching $\|\mathbf{G}\mathbf{c}^* - \mathbf{d}\|_2 \approx 130$; 20 samples for each box plot.

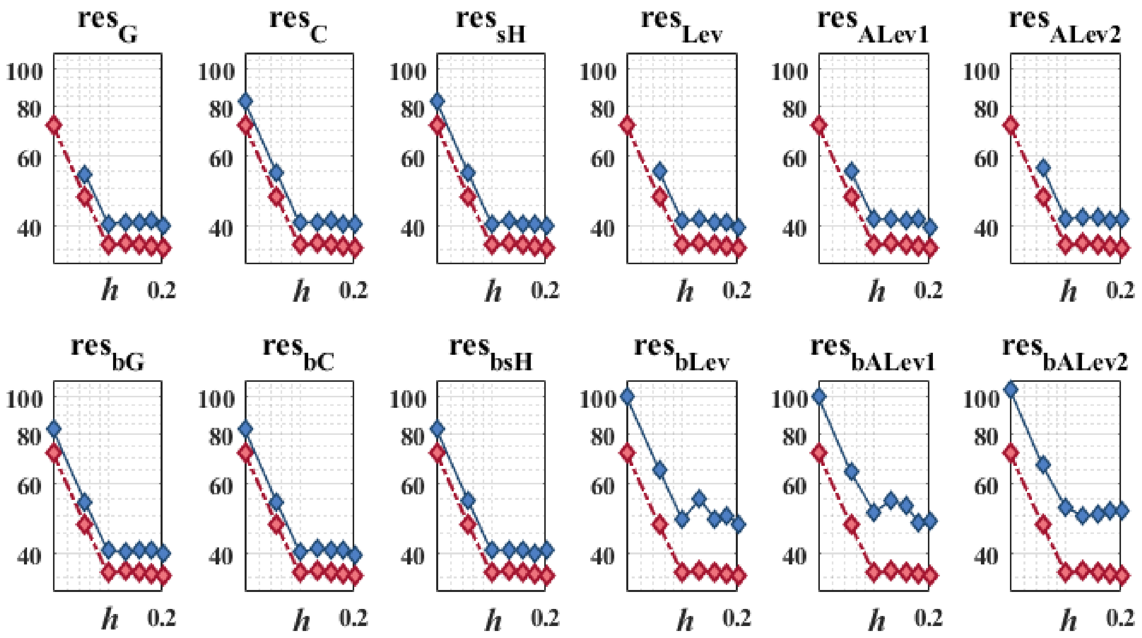

**Fig.C.2.5** Residuals after sketching $\|\mathbf{G}\hat{\mathbf{c}} - \mathbf{d}\|_2$ (blue solid diamonds), and residuals without sketching $\|\mathbf{G}\mathbf{c}^* - \mathbf{d}\|_2$ (red dashed diamonds) for the corresponding examples in Fig.5.6.3 and Fig.5.6.4 displayed in log10 scale. $h = [.050, .075, .100, .125, .150, .175, .200]$ are the same for all plots.

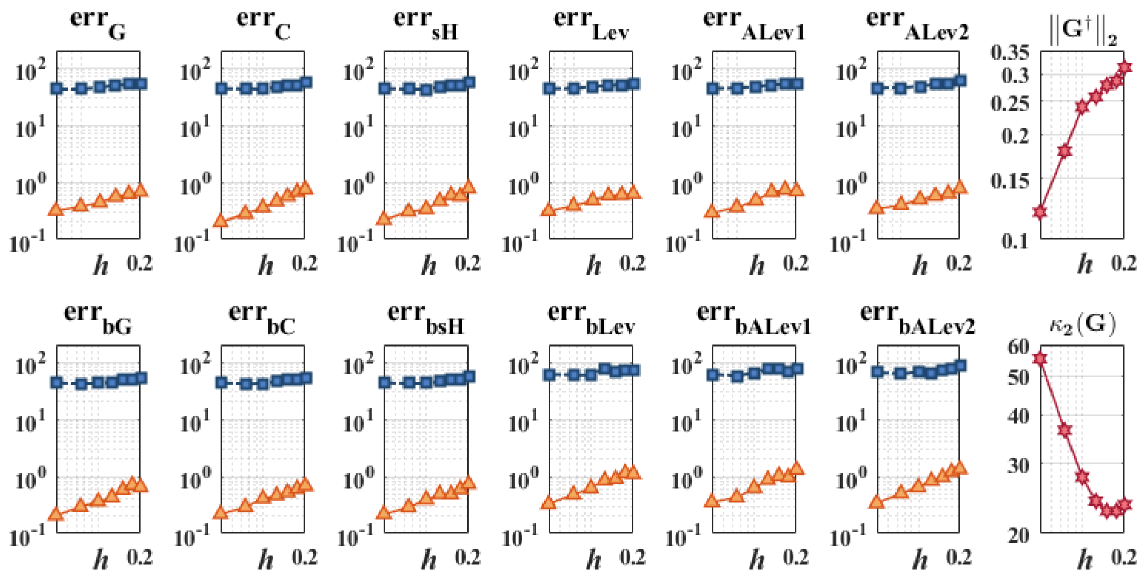

**Fig.C.2.6** Coefficient errors $\|\hat{\mathbf{c}} - \mathbf{c}^*\|_2$ (orange solid triangles), and predicted bounds for coefficient errors (blue dashed squares) for the corresponding examples in Fig.5.6.3 and Fig.5.6.4 displayed in log10 scale. $h = [.050, .075, .100, .125, .150, .175, .200]$ are the same for all plots.

# References

[1] D. Achlioptas. “Database-friendly Random Projections: Johnson-Lindenstrauss with Binary Coins”. In: *Journal of Computer and System Sciences* 66.4 (2003). Special Issue on PODS 2001, pp. 671–687. ISSN: 0022-0000. DOI: https://doi.org/10.1016/S0022-0000(03)00025-4. URL: https://www.sciencedirect.com/science/article/pii/S0022000003000254 (visited on 08/01/2024).

[2] H. Avron, P. Maymounkov, and S. Toledo. “Blendenpik: Supercharging LAPACK’s Least-Squares Solver”. In: *SIAM Journal on Scientific Computing* 32.3 (2010), pp. 1217–1236. DOI: 10.1137/090767911. URL: https://doi.org/10.1137/090767911 (visited on 07/31/2024).

[3] K. Basu. *Quasi-Monte Carlo Methods in Non-cubical Spaces*. Stanford University. 2016. URL: https://purl.stanford.edu/kn704jc4083 (visited on 07/25/2024).

[4] P.B. Bochev and M.D. Gunzburger. “Alternative Variational Formulations”. In: *Least-Squares Finite Element Methods*. Springer, 2009, pp. 35–62. URL: https://api.semanticscholar.org/CorpusID:262787461 (visited on 07/17/2024).

[5] P.B. Bochev and M.D. Gunzburger. “Classical Variational Methods”. In: *Least-Squares Finite Element Methods*. Springer, 2009, pp. 3–31. URL: https://api.semanticscholar.org/CorpusID:262787461 (visited on 07/17/2024).

[6] P.B. Bochev and M.D. Gunzburger. “Finite Element Methods of Least-Squares Type”. In: *SIAM review* 40.4 (1998), pp. 789–837. DOI: 10.1137/S0036144597321156.

[7] F. Bornemann. “Matrix Factorization”. In: *Numerical Linear Algebra: A Concise Introduction with MATLAB and Julia*. Cham: Springer International Publishing, 2018, pp. 21–37. ISBN: 978-3-319-74222-9. DOI: 10.1007/978-3-319-

74222-9_2. URL: https://doi.org/10.1007/978-3-319-74222-9_2 (visited on 08/09/2024).

[8] D. Braess. “Conforming Finite Elements”. In: *Finite Elements: Theory, Fast Solvers, and Applications in Solid Mechanics*. 3rd ed. Cambridge University Press, 2007, pp. 27–104. DOI: 10.1017/CBO9780511618635.

[9] C. Cartis, J. Fiala, and Z. Shao. “Hashing Embeddings of Optimal Dimension, with Applications to Linear Least Squares”. In: *ArXiv* abs/2105.11815 (2021). URL: https://api.semanticscholar.org/CorpusID:235187375 (visited on 07/31/2024).

[10] M.A. Celia. “Basic Concepts for Collocation”. In: *Collocation on Deformed Finite Elements and Alternating Direction Collocation Methods*. Princeton University, 1983, pp. 1–35.

[11] S. Chenakkod et al. “Optimal Embedding Dimension for Sparse Subspace Embeddings”. In: *Proceedings of the 56th Annual ACM Symposium on Theory of Computing* (2023). URL: https://api.semanticscholar.org/CorpusID:265281115 (visited on 08/01/2024).

[12] P.G. Ciarlet. “Conforming Finite Element Methods for Second-Order Problems”. In: *The Finite Element Method for Elliptic Problems*. Society for Industrial and Applied Mathematics, 2002, pp. 110–173. DOI: 10.1137/1.9780898719208.ch3. URL: https://epubs.siam.org/doi/abs/10.1137/1.9780898719208.ch3 (visited on 08/09/2024).

[13] K. Clarkson and D. Woodruff. “Low-Rank Approximation and Regression in Input Sparsity Time”. In: *Journal of the ACM* 63 (Jan. 2017), pp. 1–45. DOI: 10.1145/3019134. URL: https://arxiv.org/abs/1207.6365 (visited on 08/01/2024).

[14] M.B. Cohen. “Nearly Tight Oblivious Subspace Embeddings by Trace Inequalities”. In: *ACM-SIAM Symposium on Discrete Algorithms*. 2016. URL: https://api.semanticscholar.org/CorpusID:38839441 (visited on 08/01/2024).

[15] B.N. Datta. “QR Factorization, Singular Value Decomposition, and Projections”. In: *Numerical Linear Algebra and Applications, 2nd Edition*. 2nd ed. Philadelphia, PA: Society for Industrial and Applied Mathematics, 2010, pp. 181–

228. DOI: 10.1137/1.9780898717655. URL: https://epubs.siam.org/doi/abs/10.1137/1.9780898717655 (visited on 08/09/2024).

[16] P. Drineas et al. “Fast Approximation of Matrix Coherence and Statistical Leverage”. In: *The Journal of Machine Learning Research* 13.1 (2012), pp. 3475–3506. URL: https://www.jmlr.org/papers/volume13/drineas12a/drineas12a.pdf (visited on 07/31/2024).

[17] P. Drineas et al. “Faster least squares approximation”. In: *Numer. Math.* 117.2 (Feb. 2011), pp. 219–249. ISSN: 0029-599X. DOI: 10.1007/s00211-010-0331-6. URL: https://doi.org/10.1007/s00211-010-0331-6 (visited on 08/01/2024).

[18] E.D. Eason. “A Review of Least-Squares Methods for Solving Partial Differential Equations”. In: *International Journal for Numerical Methods in Engineering* 10.5 (1976), pp. 1021–1046. DOI: https://doi.org/10.1002/nme.1620100505.

[19] P.E. Farrell. *Lecture Notes for C6.4 Finite Element Methods for PDEs*. University of Oxford. 2021. URL: https://people.maths.ox.ac.uk/farrellp/femvideos/notes.pdf (visited on 07/01/2024).

[20] V. Girault and P.A. Raviart. “Mathematical Foundation of the Stokes Problem”. In: *Finite Element Methods for Navier-Stokes Equations: Theory and Algorithms*. Springer-Verlag, 1986, pp. 1–109.

[21] The MathWorks Inc. *Discrete Cosine Transform*. Natick, Massachusetts, United States, 2024. URL: https://uk.mathworks.com/help/signal/ref/dct.html (visited on 08/02/2024).

[22] The MathWorks Inc. *Partial Differential Equation Toolbox*. Natick, Massachusetts, United States, 2023. URL: https://ww2.mathworks.cn/products/pde.html (visited on 06/01/2024).

[23] The MathWorks Inc. *QR Decomposition*. Natick, Massachusetts, United States, 2024. URL: https://uk.mathworks.com/help/matlab/ref/qr.html (visited on 08/09/2024).

[24] The MathWorks Inc. *Wilcoxon Rank Sum Test.* Natick, Massachusetts, United States, 2024. URL: https://uk.mathworks.com/help/stats/ranksum.html (visited on 08/09/2024).

[25] B. Jiang. "Basis of LSFEM". In: *The Least-Squares Finite Element Method: Theory and Applications in Computational Fluid Dynamics and Electromagnetics.* Cambridge University Press, 1998, pp. 47–78. DOI: 10.1017/S0022112099227966.

[26] B. Jiang. "Div-Curl System". In: *The Least-Squares Finite Element Method: Theory and Applications in Computational Fluid Dynamics and Electromagnetics.* Cambridge University Press, 1998, pp. 81–95. DOI: 10.1017/S0022112099227966.

[27] B. Jiang. "Div-Curl-Grad System". In: *The Least-Squares Finite Element Method: Theory and Applications in Computational Fluid Dynamics and Electromagnetics.* Cambridge University Press, 1998, pp. 97–111. DOI: 10.1017/S0022112099227966.

[28] K. Krupchyk and J. Tuomela. "The Sharpiro–Lopatinskij Condition for Elliptic Boundary Value Problems". In: *LMS Journal of Computation and Mathematics* 9 (2006), pp. 287–329. DOI: 10.1112/S1461157000001285.

[29] B.W. Larsen and T.G. Kolda. *Sketching Matrix Least Squares via Leverage Scores Estimates.* 2022. arXiv: 2201.10638 [math.NA]. URL: https://arxiv.org/abs/2201.10638 (visited on 07/31/2024).

[30] G. Maierhofer and D. Huybrechs. *Convergence Analysis of Oversampled Collocation Boundary Element Methods in 2D.* 2021. arXiv: 2103.17212 [math.NA]. URL: https://arxiv.org/abs/2103.17212 (visited on 08/09/2024).

[31] P. Martinsson and J.A. Tropp. "Randomized numerical linear algebra: Foundations and algorithms". In: *Acta Numerica* 29 (2020), pp. 403–572. DOI: 10.1017/S0962492920000021.

[32] W.H. Mc Ewen. "On the Approximate Solution of Linear Differential Equations with Boundary Conditions". In: *Bulletin of the American Mathematical Society* 38.12 (1932), pp. 887–894. URL: https://projecteuclid.org/journals/bulletin-of-the-american-mathematical-society-new-series/volume-38/issue-12/On-the-approximate-solution-of-linear-differential-equations-with-boundary/bams/1183496405.full (visited on 07/22/2024).

[33] R.F. Millar. “The Rayleigh Hypothesis and a Related Least-Squares Solution to Scattering Problems for Periodic Surfaces and Other Scatterers”. In: *Radio Science* 8.8-9 (1973), pp. 785–796. DOI: https://doi.org/10.1029/RS008i008p00785.

[34] R. Murray et al. *Randomized Numerical Linear Algebra: A Perspective on the Field With an Eye to Software.* Tech. rep. UCB/EECS-2023-19. University of California, Berkeley, Feb. 2023. URL: http://www2.eecs.berkeley.edu/Pubs/TechRpts/2023/EECS-2023-19.html (visited on 07/31/2024).

[35] R. Osada et al. “Shape Distributions”. In: *ACM Trans. Graph.* 21.4 (Oct. 2002), pp. 807–832. ISSN: 0730-0301. DOI: 10.1145/571647.571648.

[36] Mark Rudelson and Roman Vershynin. “Non-asymptotic Theory of Random Matrices: Extreme Singular Values”. In: *arXiv: Functional Analysis* (2010). URL: https://api.semanticscholar.org/CorpusID:17320180 (visited on 08/09/2024).

[37] Z. Shao. *On Random Embeddings and Their Application to Optimisation.* 2022. arXiv: 2206.03371 [math.OC]. URL: https://arxiv.org/abs/2206.03371 (visited on 07/31/2024).

[38] V. Vijayan. *Fast SVD and PCA.* 2024. URL: https://uk.mathworks.com/matlabcentral/fileexchange/47132-fast-svd-and-pca (visited on 07/24/2024).

[39] S. Wang. “A Practical Guide to Randomized Matrix Computations with MATLAB Implementations”. In: *ArXiv* abs/1505.07570 (2015). URL: https://api.semanticscholar.org/CorpusID:1247505 (visited on 07/31/2024).

[40] G.V. Winckel. *Gaussian Quadrature for an N-dimensional Simplex.* 2005. URL: https://github.com/eschnett/SimplexQuad.jl (visited on 07/22/2024).

[41] D.P. Woodruff. “Sketching as a Tool for Numerical Linear Algebra”. In: *Foundations and Trends in Theoretical Computer Science* 10.1–2 (2014), pp. 1–157. ISSN: 1551-305X. DOI: 10.1561/0400000060. URL: http://dx.doi.org/10.1561/0400000060 (visited on 07/31/2024).

[42] K. Yasuura. “A View of Numerical Methods in Diffraction Problems”. In: *Progress in Radio Science 1966-1969* (1971), pp. 267–270.