École polytechnique fédérale de Lausanne

Semester project - fall 2025

Master in Applied Mathematics

Improving stability and performance of s-step Krylov subspace methods with randomization

Author:
Yuekai Yan

Supervisor:
Laura Grigori
Taejun Park
Igor Simunec

Contents

1 Introduction

Solving large-scale linear systems is a central task in modern numerical linear algebra and arises ubiquitously in scientific computing and engineering applications. With the increasing availability of high-performance computing (HPC) platforms and large-scale distributed systems, Krylov subspace methods have become an effective approach for solving large-scale linear systems. However, when deployed on modern compute clusters, the performance of these methods is often limited not only by floating-point operations, but also by the communication cost among computing nodes, which has emerged as a critical bottleneck.

To mitigate communication overhead, communication-avoiding techniques, most notably the s-step Krylov subspace methods [16], have been proposed. These methods reduce synchronization frequency by generating multiple Krylov vectors per iteration, thereby significantly improving parallel scalability. Despite their communication advantages, s-step methods are known to be more susceptible to numerical instability due to finite-precision arithmetic. In particular, generating s Krylov vectors simultaneously tends to amplify rounding errors, especially when the monomial basis is employed. As a result, the s Krylov vectors produced within a block may rapidly lose linear independence due to the convergence behavior of the power method, leading to near collinearity and a severely degraded Krylov subspace that hinders convergence.

Several strategies have been proposed to alleviate these numerical issues. A straightforward approach is to restrict the block size s to relatively small values, at the expense of reduced communication efficiency. Another widely studied technique is the use of alternative polynomial bases, such as the Newton basis [3; 29], which can significantly improve numerical stability compared with the monomial basis. While these approaches offer partial remedies, they do not fully address the fundamental tension between communication efficiency and numerical robustness.

In recent years, randomized numerical linear algebra (RandNLA) has emerged as a powerful framework for designing algorithms that are both computationally efficient and numerically robust. Techniques based on subspace embeddings such as the sketch-and-solve paradigm [5; 18] have demonstrated remarkable success in large-scale sparse linear algebra problems. By replacing exact algebraic operations with carefully designed randomized approximations, these methods often achieve improved stability properties while simultaneously reducing computational and communication costs.

Motivated by these developments, this work investigates the effectiveness and robustness of randomized techniques within the GMRES framework, with a particular focus on randomized orthogonalization strategies [13] in s-step methods. Specifically, we study randomized block Gram-Schmidt (RBGS) procedures and systematically compare different combinations of randomized intra-block and randomized inter-block orthogonalization schemes. Our goal is to investigate how randomization and different orthogonalization strategies influence numerical stability, orthogonality preservation, and convergence behavior in communication-avoiding GMRES. The organization of this project is as follows.

In Section 2, we introduce the notation and review fundamental concepts in numerical linear algebra relevant to this work, including Krylov subspace methods, GMRES and subspace embeddings.

Section 3 presents deterministic and randomized variants of the block Gram-Schmidt process, which serves as the foundation of the randomized s-step Krylov framework.

In Section 4, we examine several randomized intra-block and inter-block orthogonalization strategies.

Section 5 discusses the incorporation of the s-step Krylov method into GMRES.

Section 6 presents numerical experiments that evaluate the proposed methods in terms of accuracy, stability, robustness, and computational efficiency.

Finally, Section 7 concludes the project and outlines directions for future research.

All experiments were performed in MATLAB R2024a on a workstation with an Apple M4 Pro CPU (14 cores), 48 GB of unified memory, and a 512 GB solid-state drive. The source code in this project is publicly available at `https://github.com/yuekai-yan/S-step-Krylov-subspace-methods`.

2 Preliminaries

In this section we briefly go through the basic knowledge of Krylov subspace methods and random subspace embeddings, which are the basic concepts that will be referenced constantly in the paper.

2.1 Notations

We introduce here some general notation that we use throughout this project. Matrices are denoted by uppercase letters, while vectors are denoted by bold lowercase letters. We use MATLAB style notation for vectors and matrices. For example, consider a matrix A, for the i th to j th columns of A, we write $A(:, i : j)$. The Euclidean 2-norm of a vector $\mathbf{x}$ is denoted by $\|\mathbf{x}\|$, and the Frobenius norm of a matrix X is denoted by $\|X\|_F$. The transpose of a matrix X is written as X^T. We denote by $\mathbf{0}_n$ the zero vector of length n, by $\mathbf{0}_{n\times m}$ the zero matrix of size $n \times m$, and by I_n the $n \times n$ identity matrix, the subscripts may be omitted if there is no ambiguity on the dimensions. The columns of the identity matrix of dimension n are denoted by $\mathbf{e}_1, \ldots, \mathbf{e}_n$. The singular values of A, arranged in nonincreasing order, are denoted by $\sigma_1(A), \ldots, \sigma_n(A)$.

2.2 Krylov subspace methods

The Krylov subspace method is the fundamental subspace iteration methods and the important component in a number of numerical algorithms. The basic idea of Krylov subspace methods is to extract as much useful information as possible from the subspace spanned by the orbit $\mathbf{x}, A\mathbf{x}, \ldots, A^{k-1}\mathbf{x}$ of a given vector $\mathbf{x}$ under the action of a matrix A. The resulting Krylov subspace is defined as follows.

Definition 2.1 (Krylov subspace)**.** Let $A \in \mathbb{C}^{n\times n}$ and $\mathbf{x} \in \mathbb{C}^n$. Then

$$\mathcal{K}_k(A, \mathbf{x}) = \operatorname{span}\{\mathbf{x}, A\mathbf{x}, \ldots, A^{k-1}\mathbf{x}\}$$

is called the k th Krylov subspace for A and $\mathbf{x}$.

To extract a suitable basis from the Krylov subspace, an orthogonalization procedure is required. In practice, numerical stability is a central concern, as finite-precision arithmetic can severely degrade the quality of the computed basis. The Arnoldi method Algorithm 1 [2] addresses this issue by constructing an orthonormal basis of the Krylov subspace through successive orthogonalization at each iteration. Its validity is grounded in the following fundamental property of Krylov subspaces.

Proposition 2.1. *Suppose that* $\mathbf{u}_1, \ldots, \mathbf{u}_{j-1}$ *and* $\mathbf{u}_1, \ldots, \mathbf{u}_{j-1}, \mathbf{u}_j$ *are bases for* $\mathcal{K}_{j-1}(A, \mathbf{x})$ *and* $\mathcal{K}_j(A, \mathbf{x})$*, respectively. Then*

$$\mathcal{K}_{j+1}(A, \mathbf{x}) = \operatorname{span}\{\mathbf{u}_1, \ldots, \mathbf{u}_j, A\mathbf{u}_j\}.$$

Proof. Since $A^{j-1}\mathbf{x} \in \mathcal{K}_j(A, \mathbf{x}) = \operatorname{span}\{\mathbf{u}_1, \ldots, \mathbf{u}_j\}$, there are $\beta_1, \ldots, \beta_j \in \mathbb{C}$ such that

$$A^{j-1}\mathbf{x} = \beta_1\mathbf{u}_1 + \cdots + \beta_{j-1}\mathbf{u}_{j-1} + \beta_j\mathbf{u}_j.$$

Moreover we have $\beta_j \neq 0$, because $\beta_j = 0$ would imply that $A^{j-1}\mathbf{x} \in \mathcal{K}_{j-1}(A, \mathbf{x})$ and therefore $\dim \mathcal{K}_j(A, \mathbf{x}) \leq j - 1$, contradicting the assumption that $\mathcal{K}_j(A, \mathbf{x})$ has j basis elements. From

$$A^j\mathbf{x} - \beta_j A\mathbf{u}_j = A\sum_{i=1}^{j-1}\beta_i\mathbf{u}_i \in \mathcal{K}_j(A, \mathbf{x}),$$

it follows that the vectors $A^j\mathbf{x}$ and a nonzero multiple of $A\mathbf{u}_j$ only differ by an element from $\mathcal{K}_j(A, \mathbf{x})$. Hence, it does not matter whether we add $A^j\mathbf{x}$ or $A\mathbf{u}_j$ to $\mathcal{K}_j(A, \mathbf{x})$; in both cases $\mathcal{K}_{j+1}(A, \mathbf{x})$ is obtained. □

From Algorithm 1, we can obtain an important relation, which is called Arnoldi identity:

$$AU_k = U_{k+1}H_k, \tag{1}$$

where H_k is the upper Hessenberg matrix

$$H_k = \begin{pmatrix} h_{11} & h_{12} & \cdots & h_{1k} \\ h_{21} & h_{22} & \ddots & \vdots \\ & \ddots & \ddots & h_{k-1,k} \\ & & & h_{kk} \\ & & & h_{k+1,k} \end{pmatrix}.$$

Algorithm 1: Arnoldi method

Input: Matrix $A \in \mathbb{C}^{n\times n}$. Starting vector $\mathbf{x} \neq \mathbf{0}$, $k \in \mathbb{N}$.
Output: Orthonormal basis $U_{k+1} = (\mathbf{u}_1, \ldots, \mathbf{u}_k, \mathbf{u}_{k+1})$ of $\mathcal{K}_{k+1}(A, \mathbf{x})$.
Set $\mathbf{u}_1 = \mathbf{x}/\|\mathbf{x}\|$, $U_1 = \mathbf{u}_1$;
for $j = 1, 2, \ldots, k$ **do**
  Compute $\mathbf{w} = A\mathbf{u}_j$;
  Compute $\mathbf{h}_j = U_j^*\mathbf{w}$;
  Compute $\tilde{\mathbf{u}}_{j+1} = \mathbf{w} - U_j\mathbf{h}_j$;
  Set $h_{j+1,j} = \|\tilde{\mathbf{u}}_{j+1}\|$;
  Set $\mathbf{u}_{j+1} = \tilde{\mathbf{u}}_{j+1}/h_{j+1,j}$;
  Set $U_{j+1} = (U_j, \mathbf{u}_{j+1})$;
end

The Arnoldi identity plays a central role in extracting useful information from the Krylov subspace, for example in approximating eigenvalues and eigenvectors of A or in solving linear systems $A\mathbf{x} = \mathbf{b}$ via GMRES [22] or the Full Orthogonalization Method (FOM) [2; 21]. By reducing the large-scale matrix A to a much smaller Hessenberg matrix H while approximately preserving the spectral information of A, it enables efficient and numerically tractable computations.

2.3 GMRES and s-step GMRES

Given a linear system

$$A\mathbf{x} = \mathbf{b},$$

and starting vector $\mathbf{x}_0$ (e.g., $\mathbf{x}_0 = 0$), a number of popular iterative methods extract an approximation to the solution from the Krylov subspace $\mathcal{K}_k(A, \mathbf{r}_0)$. Here, $\mathbf{r}_0 := \mathbf{b} - A\mathbf{x}_0$ denotes the residual belonging to the starting vector, from Definition 2.1, we have

$$\mathcal{K}_k(A, \mathbf{r}_0) = \operatorname{span}\{\mathbf{r}_0, A\mathbf{r}_0, \ldots, A^{k-1}\mathbf{r}_0\}.$$

There are many different ways of extracting such an approximation, each of which gives rise to a different method. The generalized minimum residual method (GMRES) [22] selects the vector from $\mathcal{K}_k(A, \mathbf{r}_0)$ that minimizes the 2-norm of the residual. For this purpose, Algorithm 1 is used to generate an orthonormal basis $U_k = (\mathbf{u}_1, \ldots, \mathbf{u}_k)$ of $\mathcal{K}_k(A, r_0)$, where $\mathbf{u}_1 = \mathbf{r}_0/\beta_0$ and $\beta_0 := \|\mathbf{r}_0\|$.

By Equation (1), the coefficients generated by the Arnoldi method can be arranged in an Arnoldi decomposition

$$AU_k = U_{k+1}H_k$$

with an upper Hessenberg matrix H_k. For a vector $\tilde{\mathbf{x}} = \mathbf{x}_0 + U_k\mathbf{y} \in \mathbf{x}_0 + \mathcal{K}_k(A, \mathbf{r}_0)$ with some $\mathbf{y} \in \mathbb{R}^k$, the 2-norm of the residual can be computed as follows:

$$\begin{aligned}\|\mathbf{b} - A\tilde{\mathbf{x}}\| &= \|\mathbf{r}_0 - AU_k\mathbf{y}\| = \|\mathbf{r}_0 - U_{k+1}H_k\mathbf{y}\| \\ &= \|U_{k+1}(\beta_0\mathbf{e}_1 - H_k\mathbf{y})\| = \|\beta_0\mathbf{e}_1 - H_k\mathbf{y}\|.\end{aligned}$$

Hence, the residual norm is minimized when $\mathbf{y}$ solves the $(k+1)\times k$ linear least-squares problem

$$\min_{\mathbf{y}\in\mathbb{R}^k}\|\beta_0\mathbf{e}_1 - H_k\mathbf{y}\|. \tag{2}$$

If we let $\mathbf{y}_k$ denote the solution of Equation (2), then the solution produced after k steps of GMRES is given by

$$\mathbf{x}_k := \mathbf{x}_0 + U_k\mathbf{y}_k.$$

The s-step GMRES method is a communication-avoiding variant of GMRES designed to reduce the synchronization and communication costs incurred by the Arnoldi process. Instead of expanding the Krylov subspace one vector at a time, s-step GMRES generates a block of s Krylov basis vectors in each outer iteration, thereby constructing a larger Krylov subspace using fewer global reductions. In exact arithmetic, s-step GMRES produces the same approximation as standard GMRES after the same total number of Krylov basis vectors have been generated; however, in finite-precision arithmetic, the numerical behavior can differ.

2.4 Subspace embeddings

Randomized subspace embedding can be viewed as a technique of dimension reduction and is widely used in computational linear algebra [28]. Its central idea is to embed a high-dimensional problem into a lower space while approximately preserving the geometric property, which means approximately preserving the inner product of the sketched vector,

$$\langle \cdot, \cdot \rangle \approx \langle \Theta \cdot, \Theta \cdot \rangle.$$

These random linear maps preserve sufficient information to enable, with high probability, the accurate solution of a wide range of linear algebra problems [1; 10; 12; 17]. The definition below follows [23],

Definition 2.2. For $\epsilon < 1$, the sketching matrix $\Theta \in \mathbb{R}^{d \times n}$ is said to be an ϵ-subspace embedding for $\mathcal{V} \subseteq \mathbb{R}^n$ if we have

$$|\langle x, y \rangle - \langle \Theta x, \Theta y \rangle| \leq \epsilon \|x\| \|y\| \quad \forall x, y \in \mathcal{V}. \tag{3}$$

The subspace embedding property can also be interpreted from several alternative perspectives, which provide useful insights for both the construction of subspace embeddings and their applications to numerical linear algebra problems. The next proposition provides six equivalent characterizations of the notion of ϵ-subspace embedding.

Proposition 2.2. *Suppose that $\mathcal{V} = \text{range}(V)$ where $V \in \mathbb{R}^{n \times m}$ is a matrix with orthonormal columns. For a matrix $\Theta \in \mathbb{R}^{d \times n}$, the following are equivalent:*

(i) Θ *is an ϵ-subspace embedding for $\mathcal{V}$*

(ii) $(1-\epsilon)\|\mathbf{v}\|^2 \leq \|\Theta \mathbf{v}\|^2 \leq (1+\epsilon)\|\mathbf{v}\|^2, \qquad \forall\, \mathbf{v} \in \mathcal{V}$

(iii) $\max\left\{ \left| \|\Theta \mathbf{v}\|^2 - 1 \right| : \mathbf{v} \in \mathcal{V},\ \|\mathbf{v}\| = 1 \right\} \leq \epsilon$

(iv) $\sigma^2_{\min}(\Theta V) \geq 1 - \epsilon, \qquad \sigma^2_{\max}(\Theta V) \leq 1 + \epsilon$

(v) $\lambda_{\min}\left(V^\mathsf{T} \Theta^\mathsf{T} \Theta V\right) \geq 1 - \epsilon, \quad \lambda_{\max}\left(V^\mathsf{T} \Theta^\mathsf{T} \Theta V\right) \leq 1 + \epsilon$

(vi) $\|I - V^\mathsf{T} \Theta^\mathsf{T} \Theta V\| \leq \epsilon$

Proof. The implication (i) $\Rightarrow$ (ii) follows readily from replacing x and y in Equation (3) by an arbitrary $\mathbf{v} \in \mathcal{V}$.

Next, let us prove (ii) $\Rightarrow$ (i). Since

$$(1-\epsilon)\|\mathbf{v}\|^2 \leq \|\Theta \mathbf{v}\|^2 \leq (1+\epsilon)\|\mathbf{v}\|^2, \qquad \forall\, \mathbf{v} \in \mathcal{V}$$

we have,

$$-\epsilon \|\mathbf{v}\|^2 \leq \langle \mathbf{v}, \left(I - \Theta^\star \Theta\right) \mathbf{v} \rangle \leq \epsilon \|\mathbf{v}\|^2, \qquad \forall\, \mathbf{v} \in \mathcal{V}.$$

Since $\Theta^\star \Theta - I$ is a symmetric operator on $\mathbb{R}^n$, we deduce that

$$\|I - \Theta^\star \Theta\| \leq \epsilon.$$

Therefore,

$$|\langle \mathbf{x}, \mathbf{y} \rangle - \langle \Theta \mathbf{x}, \Theta \mathbf{y} \rangle| = |\langle \mathbf{x}, \left(I - \Theta^\star \Theta\right) \mathbf{y} \rangle| \leq \|I - \Theta^\star \Theta\| \cdot \|\mathbf{x}\| \|\mathbf{y}\| \leq \epsilon \|\mathbf{x}\| \|\mathbf{y}\|.$$

The equivalence of (ii) and (iii) follows by normalizing v.

To prove (ii) $\Rightarrow$ (iv), $\forall \mathbf{v} \in \mathcal{V}$ can be represented by Vx for some $x \in \mathbb{R}^m$, then it follows that

$$(1-\epsilon)\|\mathbf{x}\|^2 \leq \|\Theta V \mathbf{x}\|^2 \leq (1+\epsilon)\|\mathbf{x}\|^2, \qquad \forall\, \mathbf{x} \in \mathbb{R}^m.$$

By the definition of singular value, we prove the equivalence of (ii) and (iv).

Then the equivalence of (ii) and (iii) follows immediately by the definition of singular value. And the equivalence of (v) and (vi) follows immediately by

$$\|I - V^\mathsf{T} \Theta^\mathsf{T} \Theta V\| = max\{\left|\lambda_{min}\left(I - V^\mathsf{T} \Theta^\mathsf{T} \Theta V\right)\right|, \left|\lambda_{max}\left(I - V^\mathsf{T} \Theta^\mathsf{T} \Theta V\right)\right|\} \leq \epsilon.$$

□

By the subspace embedding property, we obtain the following useful relations, which establish the spectral relationship between the matrix V and its sketched counterpart ΘV, as well as the corresponding relationship between their condition numbers [15]:

$$\sqrt{1-\epsilon} \leq \frac{\sigma_j(\Theta V)}{\sigma_j(V)} \leq \sqrt{1+\epsilon}\,, \tag{4}$$

$$Cond(V) \leq \frac{\sqrt{1+\epsilon}}{\sqrt{1-\epsilon}}\, Cond(\Theta V)\,. \tag{5}$$

The definition of the ϵ-subspace embedding also provides a good approximation for a least-square problem by solving the sketched problem [14; 20]:

$$min\{\|\Theta A\tilde{\mathbf{x}} - \Theta\mathbf{b}\| : \tilde{\mathbf{x}} \in \mathbb{R}^m\}. \tag{6}$$

instead of solving $min\{\|A\mathbf{x} - \mathbf{b}\| : \mathbf{x} \in \mathbb{R}^m\}$.

Lemma 2.1 (sketch and solve [23])**.** *Suppose that* $\tilde{\mathbf{x}}$ *solves the sketched least-squares problem Equation* (6) *with an* ϵ*-subspace embedding* Θ *of the subspace span* $([A, \mathbf{b}])$ *and* $\mathbf{x}$ *is the real solution of the original least-square problem. Then*

$$\|A\tilde{\mathbf{x}} - \mathbf{b}\|^2 \leq \frac{1+\epsilon}{1-\epsilon}\|A\mathbf{x} - \mathbf{b}\|^2.$$

Proof. Since $A\mathbf{x} - \mathbf{b}, A\tilde{\mathbf{x}} - \mathbf{b} \in span\,([A, \mathbf{b}])$, we have

$$\begin{aligned}
\|A\tilde{\mathbf{x}} - \mathbf{b}\|^2 &\leq \frac{1}{1-\epsilon}\|\Theta A\tilde{\mathbf{x}} - \Theta\mathbf{b}\|^2 \\
&\leq \frac{1}{1-\epsilon}\|\Theta A\mathbf{x} - \Theta\mathbf{b}\|^2 \\
&\leq \frac{1+\epsilon}{1-\epsilon}\|A\mathbf{x} - \mathbf{b}\|^2.
\end{aligned}$$

□

2.4.1 Oblivious Subspace Embedding (OSE)

A trivial subspace embedding for the subspace $\mathcal{V} \subseteq \mathbb{R}^n$ is to take $\Theta = V^\mathsf{T}$, however, we aim for cheap constructions that use little or even no information about the subspace $\mathcal{V}$, which introduces Definition 2.3.

Definition 2.3 ((k, ϵ, δ)-Oblivious Subspace Embedding (OSE))**.** A (k, ϵ, δ)-oblivious subspace embedding (OSE) is a random matrix Θ such that, for a fixed but arbitrary k-dimensional subspace $\mathcal{V} \subset \mathbb{R}^n$, the matrix Θ is an ϵ-subspace embedding with probability at least $1 - \delta$. That is,

$$(1-\epsilon)\,\|\mathbf{v}\|^2 \;\leq\; \|\Theta\mathbf{v}\|^2 \;\leq\; (1+\epsilon)\,\|\mathbf{v}\|^2, \qquad \forall\, \mathbf{v} \in \mathcal{V}.$$

For $k = 1$, OSE reduces to Johnson-Lindenstrauss embedding [17]:

Definition 2.4 ((ϵ, δ)-Johnson–Lindenstrauss (JL) property)**.** A random matrix Θ satisfies the (ϵ, δ)-Johnson–Lindenstrauss (JL) property if for a fixed but arbitrary vector $v \in \mathbb{R}^n$ the inequalities

$$(1-\epsilon)\,\|\mathbf{v}\|^2 \;\leq\; \|\Theta\mathbf{v}\|^2 \;\leq\; (1+\epsilon)\,\|\mathbf{v}\|^2$$

hold with probability at least $1 - \delta$.

To extend JL property to OSE for an arbitraty k-dimensional subspace $\mathcal{V} \subset \mathbb{R}^n$, the union bound suffers from the problem that $\mathcal{V}$ contains infinitely many vectors. We can employ the idea of ϵ-net from stochastic analysis to deal with such problem:

Lemma 2.2 (Lemma 5.2 in [25])**.** *Let* S^{k-1} *be the unit sphere in* $\mathbb{R}^k$*,* S^{k-1} *has an* ϵ*-net* $\mathcal{N}_\epsilon$ *of cardinality at most* $(1 + 2/\epsilon)^k$*, that is, for every* $\mathbf{x} \in S^{k-1}$ *there is* $\mathbf{y} \in \mathcal{N}_\epsilon$ *such that*

$$\|\mathbf{x} - \mathbf{y}\| \leq \epsilon\,.$$

Moreover, we have the following corollary, which bounds the operator norm by considering the points on the ϵ-net :

Corollary 2.1. *Let $C \in \mathbb{R}^{k\times k}$ be a symmetric matrix and $\mathcal{N}_\epsilon$ be ϵ-net from Lemma 2.2, then*

$$\|C\| \leq (1-2\epsilon)^{-1}\, max\{|\mathbf{y}^\mathsf{T} C\mathbf{y}| : \mathbf{y} \in \mathcal{N}_\epsilon\}$$

Proof. Let $x \in S^{k-1}$ s.t. $\|C\| = |x^\mathsf{T} Cx|$ and let $\mathbf{y} \in \mathcal{N}_\epsilon$ s.t. $\|x-\mathbf{y}\| \leq \epsilon$. Then

$$\begin{aligned}\left|\mathbf{x}^\mathsf{T} C\mathbf{x} - \mathbf{y}^\mathsf{T} C\mathbf{y}\right| &= \left|\mathbf{x}^\mathsf{T} C\,(\mathbf{x}-\mathbf{y}) + (\mathbf{x}-\mathbf{y})^\mathsf{T} C\mathbf{y}\right| \\ &\leq \|C\|\cdot\|\mathbf{x}\|\cdot\|\mathbf{x}-\mathbf{y}\| + \|C\|\cdot\|\mathbf{y}\|\cdot\|\mathbf{x}-\mathbf{y}\| \\ &\leq 2\epsilon\|C\|.\end{aligned}$$

This implies

$$\left|\mathbf{y}^\mathsf{T} C\mathbf{y}\right| \geq \left|\mathbf{x}^\mathsf{T} C\mathbf{x}\right| - \left|\mathbf{x}^\mathsf{T} C\mathbf{x} - \mathbf{y}^\mathsf{T} C\mathbf{y}\right| \geq (1-2\epsilon)\,\|C\|.$$

□

By Lemma 2.2 and Corollary 2.1, we can thus extend JL to OSE and get the following Theorem 2.1.

Theorem 2.1. *Let $\mathcal{V} \subseteq \mathbb{R}^n$ be a fixed k-dimensional subspace, and let $V \in \mathbb{R}^{n\times k}$ denote the matrix whose columns form an orthonormal basis of $\mathcal{V}$. Any random matrix Θ satisfying the $(\epsilon/2,\ \delta/9^k)$-Johnson–Lindenstrauss (JL) property also satisfies the (k,ϵ,δ)-oblivious subspace embedding (OSE) property.*

Proof. Since every normalized vector in $\mathcal{V}$ can be written as $V\mathbf{y}, \mathbf{y} \in S^{k-1}$, by JL property, we have for $\forall\ \mathbf{y} \in S^{k-1}$,

$$\left|\mathbf{y}^\mathsf{T}\left(I - V^\mathsf{T}\Theta^\mathsf{T}\Theta V\right)\mathbf{y}\right| \leq \epsilon/2$$

with probability $\geq 1-\delta/9^k$. Take $\epsilon = 1/4$, by Lemma 2.2, we deduce that there are at most 9^k points in $\mathcal{N}_{1/4} \subset S^{k-1}$, thus we have,

$$P\left(\left|\mathbf{y}^\mathsf{T}\left(I - V^\mathsf{T}\Theta^\mathsf{T}\Theta V\right)\mathbf{y}\right| \leq \epsilon/2,\ \forall\ \mathbf{y} \in \mathcal{N}_{1/4}\right) \geq 1-\delta,$$

then by Corollary 2.1,

$$\|I - V^\mathsf{T}\Theta^\mathsf{T}\Theta V\| \leq 2max\{\left|\mathbf{y}^\mathsf{T}\left(I - V^\mathsf{T}\Theta^\mathsf{T}\Theta V\right)\mathbf{y}\right| : \mathbf{y} \in \mathcal{N}_{1/4}\} \leq \epsilon,$$

has probability at least $1-\delta$, which shows OSE.

□

2.4.2 Gaussian embeddings

The Gaussian embedding is the most basic subspace embedding. Consider a standard Gaussian random matrix with independent and identically distributed *i.i.d.* standard normal entries:

$$\Omega \in \mathbb{R}^{d\times n} \text{ with i.i.d. } \mathcal{N}(0,1) \text{ entries.}$$

Since any linear combination of independent standard Gaussian random variables is Gaussian, for any $\mathbf{x} \in \mathbb{R}^n$ with $\|\mathbf{x}\| = 1$, we have,

$$\Omega\mathbf{x} \in \mathbb{R}^d \text{ with i.i.d. } \mathcal{N}(0,1) \text{ entries.}$$

Let

$$Y := \frac{\|\Omega\mathbf{v}\|^2}{\|\mathbf{v}\|^2} = \sum_{i=1}^{n} \langle \Omega_i, \frac{\mathbf{v}}{\|\mathbf{v}\|}\rangle^2$$

where Θ_i is the ith row of Θ. Since $\langle \Omega_i, \frac{\mathbf{v}}{\|\mathbf{v}\|}\rangle \sim \mathcal{N}(0,1)$ are independent, we deduce that $Y \sim \chi^2_n$. The following inequality follows from the tail bound for the chi-squared distribution [26]:

$$\mathbb{P}\left\{\left|\frac{\|\Omega\mathbf{v}\|^2}{d\,\|\mathbf{v}\|^2} - 1\right| \geq \epsilon\right\} \ \leq\ 2\exp\left(-\frac{d\epsilon^2}{8}\right), \qquad 0 \leq \epsilon < 1.$$

Let $\Theta_G = \frac{1}{\sqrt{d}}\Omega$. Combining the tail bound with Theorem 2.1, in order to achieve the (m,ϵ,δ)-OSE property for an arbitrary m-dimensional subspace $\mathcal{V} \subset \mathbb{R}^n$, the embedding dimension must satisfy $d \geq O\left(\epsilon^{-2}\,(m + \log\frac{1}{\delta})\right)$ [7]. In practice, a common and convenient choice is $d = 2m$.

2.4.3 Rademacher

A Rademacher matrix $R \in \mathbb{R}^{d\times n}$ is a random matrix with i.i.d. entries R_{ij} :

$$P(R_{ij} = 1) = \frac{1}{2}, \quad P(R_{ij} = -1) = \frac{1}{2}, \quad i = 1, \ldots, d, \quad j = 1, \ldots, n.$$

Let $\Theta_R = \frac{1}{\sqrt{d}}R$. Similar to the Θ_G, we can prove that to achieve the OSE property, the embedding dimension must satisfy $d \geq O\Big(\epsilon^{-2}\,(m + \log\frac{1}{\delta})\Big)$ [7; 23; 27].

2.4.4 Sparse sign embeddings

The sparse sign embedding is a sparse matrix with nonzero entries that are random signs [19] has the following form

$$\mathbf{\Theta}_S = \frac{1}{\sqrt{\xi}}\,[s_1, s_2, \ldots, s_n] \;\in\; \mathbb{R}^{d\times n}, \tag{7}$$

where the columns s_i of Θ_S are i.i.d. sparse vectors with exactly ξ nonzero entries whose locations are chosen uniformly at random, and whose nonzero values are i.i.d. Rademacher random variables. Here, ξ is the sparsity parameter. When $\xi = 1$, the sparse sign embedding is also refered to as the CountSketch matrix.

To ensure a sparse sign embedding to be oblivious for a matrix $A \in \mathbb{R}^{n\times m}$, when $\xi = 1$, OSE holds for $d = O(m^2\epsilon^{-2}\delta^{-1})$ [19]. The sparsity parameter is reasonable to be chosen as $\xi = O\left(\epsilon^{-1}\left(\log m + \log\delta^{-1}\right)\right)$ and OSE holds for $d = O\left(\epsilon^{-2}m\left(\log m + \log\delta^{-1}\right)\right)$ [11]. In practice, $d = O(m)$ and $\xi = min\{d, 8\}$ is recommended by [24].

When applying these subspace embeddings to a matrix $A \in \mathbb{R}^{n\times m}$, the cost of both Gaussian and Rademacher embeddings is $O(dnm)$, while the sparse sign embedding requires only $O(\xi nm)$. Therefore, sparse sign embeddings are often the fastest to apply in practice, especially when $\xi \ll d$.

Compared to Gaussian or Rademacher embeddings, sparse sign embeddings typically come with weaker theoretical guarantees as we usually set $\xi = 8$ instead of $\xi = O\left(\epsilon^{-1}\left(\log m + \log\delta^{-1}\right)\right)$ and $d = O(m)$ rather than $d = O(m\log m)$ in the numerical experiments. Nevertheless, sparse sign embeddings have significantly lower computational and storage costs, thereby reducing both memory usage and the cost of matrix-vector multiplication. This advantage is particularly important in large-scale settings. As a result, sparse sign embeddings offer a favorable trade-off between theoretical optimality and practical efficiency.

3 BGS Process and RBGS Process

The *Block Gram–Schmidt (BGS)* process is a block variant of the standard Gram–Schmidt orthogonalization method [8], in which multiple vectors are orthogonalized simultaneously in blocks. Compared with the classical vector-wise formulation, BGS reorganizes the orthogonalization procedure to operate on matrix blocks, thereby reducing the frequency of global synchronization.

From a parallel computing perspective, BGS can be interpreted as a *communication-avoiding* orthogonalization strategy, since inner products and projections are aggregated at the block level rather than performed for individual vectors. This significantly reduces data movement and synchronization overhead, making BGS particularly well suited for large-scale parallel and high-performance computing (HPC) environments, where communication costs often dominate arithmetic operations. This block-oriented and communication-avoiding structure makes the BGS process a natural building block for s-step Krylov subspace methods. In particular, s-step GMRES constructs and orthogonalizes multiple Krylov basis vectors per iteration, and thus requires block orthogonalization procedures that can efficiently operate on groups of vectors.

Given a matrix $X \in \mathbb{R}^{n\times m}$, the BGS process orthogonalizes the columns of X block by block. Specifically, the matrix X is partitioned into p blocks of equal size s, with $m = p \times s$,

$$X = \left[X_{(1)}, X_{(2)}, \ldots, X_{(p)}\right],$$

where each block $X_{(i)} \in \mathbb{R}^{n\times s}$. At the i-th step of the algorithm, the current block $X_{(i)}$ is first orthogonalized against all previously computed blocks $\{Q_{(1)}, \ldots, Q_{(i-1)}\}$, and then internally orthonormalized to produce a new block $Q_{(i)}$. This procedure yields a block-orthonormal matrix $Q = [Q_{(1)}, \ldots, Q_{(p)}]$, whose columns, like those produced by the standard Gram–Schmidt process, are orthonormal, and an associated upper triangular matrix R, forming a QR factorization of X, which is depicted in Algorithm 2.

The BGS process serves as a fundamental building block in block Krylov subspace methods, such as block Arnoldi and block GMRES, and plays a central role in communication-avoiding and s-step Krylov algorithms. However, similar to its classical counterpart, BGS may suffer from loss of orthogonality in finite-precision arithmetic, especially for large block sizes or ill-conditioned input matrices, motivating the study of modified and randomized variants of block orthogonalization.

Algorithm 2: BGS Process

Input: Block matrix $X = \left[X_{(1)}, X_{(2)}, \ldots, X_{(p)}\right] \in \mathbb{R}^{n\times m}$ with $m \le n$. Block size s.
Output: Orthonormal factor $Q \in \mathbb{R}^{n\times m}$ and upper triangular factor $R \in \mathbb{R}^{m\times m}$ such that $X = QR$.
for $i = 1$ **to** p **do**
  Inter-block orthogonalization:
  orthogonalize $X_{(i)}$ against blocks $Q_{(1)}, Q_{(2)}, \ldots, Q_{(i-1)}$ yielding $R(1:(i-1)s, (i-1)s+1:is)$ and $Q_i = X_{(i)} - Q(:, 1:(i-1)s)\, R(1:(i-1)s, (i-1)s:is)$;
  Intra-block orthogonalization:
  apply orthogonalization process to Q_i, yielding $Q_{(i)} = Q(:, (i-1)s+1:is)$ and $R((i-1)s+1:is, (i-1)s+1:is)$;
end

Different from its deterministic counterpart, the randomized block *Gram–Schmidt (RBGS)* algorithm introduces randomization into both the intra-block and inter-block orthogonalization procedures. The key idea is to replace exact inner products with their randomized approximations obtained via a suitably chosen sketching operator.

Specifically, RBGS applies a random sketch $\Theta \in \mathbb{R}^{d\times n}$ that satisfies the *oblivious subspace embedding* (OSE) property to the input matrix X. The algorithm then performs the Gram–Schmidt process on the sketched matrix ΘX, producing an upper-triangular factor R and a sketch-orthonormal matrix Q (i.e., $(\Theta Q)^T(\Theta Q) = I$) such that

$$X = QR,$$

In essence, RBGS orthogonalizes ΘX rather than X itself, while still forming a valid factorization of the original matrix, the RBGS process is depicted in Algorithm 3.

Owing to the OSE property of Θ, the conditioning of the factor Q can be rigorously controlled. From Equation (5), it follows that

$$Cond(Q) \le \frac{\sqrt{1+\epsilon}}{\sqrt{1-\epsilon}}\, Cond(\Theta Q) = \frac{\sqrt{1+\epsilon}}{\sqrt{1-\epsilon}},$$

which guarantees that Q is well conditioned, although not exactly orthonormal in the Euclidean inner product. This bounded conditioning plays a fundamental role in preventing the rapid loss of linear independence that commonly arises in block and communication-avoiding orthogonalization schemes, thereby enabling the construction of a numerically stable basis.

From a computational perspective, sketching the matrix X into a lower-dimensional space substantially reduces the cost of inner products and projection operations. Since these operations dominate both computation and communication in large-scale Krylov subspace methods, RBGS has the potential to significantly lower overall runtime. As a result, randomized block orthogonalization is particularly attractive in parallel and communication-avoiding settings, where the cost of global reductions often exceeds that of floating-point arithmetic [6; 18]. Beyond the computational benefits, the numerical experiments in this project further demonstrate that randomized methods lead to improved numerical stability.

Algorithm 3: RBGS Process

Input: Block matrix $X = \left[X_{(1)}, X_{(2)}, \ldots, X_{(p)}\right] \in \mathbb{R}^{n\times m}$ with $m \leq n$. Random sketch $\Theta \in \mathbb{R}^{d\times n}$. Block size s.

Output: Θ-orthonormal factor $Q \in \mathbb{R}^{n\times m}$ and upper triangular factor $R \in \mathbb{R}^{m\times m}$ such that $X = QR$.

for $i = 1$ **to** p **do**

Inter-block Θ-orthogonalization:

compute sketched matrix ΘX_i, orthogonalize ΘX_i with $S(:, 1 : (i-1)s)$, yielding $R(1 : (i-1)s, (i-1)s : is)$; compute Θ-orthogonalized matrix against blocks $Q_{(1)}, Q_{(2)}, \ldots, Q_{(i-1)}$: $Q_i = X_i - Q(:, 1 : (i-1)s)R(1 : (i-1)s, (i-1)s : is)$;

Intra-block Θ-orthogonalization:

apply randomized orthogonalization process to Q_i, yielding $Q_{(i)} = Q(:, (i-1)s + 1 : is)$ and $R((i-1)s + 1 : is, (i-1)s + 1 : is)$;

Compute sketched matrix $S(:, (i-1)s + 1 : is) = \Theta Q(:, (i-1)s + 1 : is)$;

end

4 Randomized Orthogonalization Methods

Orthogonalization methods are fundamental to all Krylov subspace algorithms, since maintaining a well-orthogonalized basis is crucial for accurately approximating the exact solution.

In this project, all randomized variants are built upon QR factorizations. Depending on how the upper triangular matrix R is computed, some standard orthogonalization procedures include Classical Gram-Schmidt (CGS), Modified Gram-Schmidt (MGS), Classical Gram-Schmidt with reorthogonalization (CGS2), their sketched counterparts are randomized CGS (rCGS), randomized MGS (rMGS), and randomized classical Gram-Schmidt with reorthogonalization (rCGS2), respectively. In this paper, we also consider Randomized Gram-Schmidt (RGS) and Randomized Cholesky QR (rCholesky).

4.1 Fundamental structure

In this subsection, we present the fundamental structure of the randomized intra-block and inter-block orthogonalization methods.

4.1.1 Intra-block

For the intra-block orthogonalization, we first compute the exact upper triangular factor R of the sketched matrix, then we use R to compute the approximate orthonormal factor Q, which is also called sketch-orthonormal or Θ-orthonormal. The corresponding procedure is summarized in Algorithm 4.

Algorithm 4: Randomized intra-block orthogonalization

Input: Matrix $X \in \mathbb{R}^{n\times m}$ with $m \le n$. Random sketch $\Theta \in \mathbb{R}^{d\times n}$.
Output: Θ-orthonormal factor $Q \in \mathbb{R}^{n\times m}$ and upper triangular factor $R \in \mathbb{R}^{m\times m}$ such that $X = QR$.
Compute the sketched matrix of X: $P = \Theta X$;
Compute the upper triangular matrix R from the QR factorization of P ;
Compute $Q = [\mathbf{q}_1, \mathbf{q}_2, \ldots, \mathbf{q}_n]$ by using R as the approximated upper triangular factor of X ;
return $Q = [\mathbf{q}_1, \ldots, \mathbf{q}_m]$ *and* R

4.1.2 Inter-block

The inter-block orthogonalization step aims to Θ-orthogonalize each column of the current block with respect to the previously Θ-orthonormalized blocks. This is achieved by first computing the corresponding coefficient matrix R, whose columns contain the Θ-inner products between the columns of the current block and the basis vectors from the preceding blocks. The Θ-orthonormal factor Q for the current block is then obtained by removing these components using the matrix R. To avoid repeated matrix multiplications and thereby reduce computational overhead, the sketched matrix is provided as an additional input. The corresponding procedure is summarized in Algorithm 5.

Algorithm 5: Randomized inter-block orthogonalization

Input: Matrix $X \in \mathbb{R}^{n\times m}$ with $m \le n$. Sketched matrix $S = \Theta X \in \mathbb{R}^{d\times m}$. Small block matrix $B \in \mathbb{R}^{n\times s}$ to be Θ-orthogonalized with X. Random sketch $\Theta \in \mathbb{R}^{d\times n}$.
Output: Θ-orthogonalized matrix $Q \in \mathbb{R}^{n\times s}$ against X and factor $R \in \mathbb{R}^{m\times s}$.
Compute the sketched matrix of X: $P = \Theta B$;
Compute R from the sketched matrix S and P ;
Compute $Q = B - XR$;
return Q *and* R

4.2 Computation of the upper triangular factor R

4.2.1 The standard Gram-Schmidt processes and their randomized variants

Given a matrix $X = [\,\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\,] \in \mathbb{R}^{n\times m}$, the standard Gram–Schmidt procedure constructs an orthonormal basis for the column space of $A \in \mathbb{R}^{m\times n}$ by sequentially orthogonalizing each column $\mathbf{x}_i$ against the previously

computed orthonormal vectors $\mathbf{q}_1, \mathbf{q}_2, \ldots, \mathbf{q}_{i-1}$. By evaluating the projection coefficients of $\mathbf{x}_i$ onto these vectors, the algorithm forms the entries of an upper triangular matrix R, while the normalized residuals define the orthonormal columns of $Q = [\,\mathbf{q}_1, \mathbf{q}_2, \ldots, \mathbf{q}_n\,]$. This procedure corresponds to the process of QR decomposition.

In more detail, at the i-th step of the classical Gram–Schmidt procedure, the vector $\mathbf{x}_i$ is decomposed into a component lying in the span of the previously constructed basis Q_{i-1} and a residual component orthogonal to it. This is achieved by subtracting from $\mathbf{x}_i$ its orthogonal projection onto Q_{i-1}, namely,

$$\tilde{\mathbf{q}}_i = \mathbf{x}_i - \sum_{j=1}^{i-1} \langle \mathbf{q}_j, \mathbf{x}_i \rangle \mathbf{q}_j.$$

The resulting vector $\tilde{\mathbf{q}}_i$ is then normalized to produce the next orthonormal basis vector $\mathbf{q}_i = \tilde{\mathbf{q}}_i / \|\tilde{\mathbf{q}}_i\|$. Equivalently, this step can be interpreted as applying an orthogonal projection operator that removes the component of $\mathbf{x}_i$ in the subspace spanned by Q_{i-1}.

The projection operator Π^{i-1} aims at projecting each column a_i of A into the orthogonal complement of Q_{i-1}. In classical Gram-Schmidt, Π^j is computed as

$$\Pi^{(j)} = I_{n\times n} - Q_j Q_j^{\mathrm{T}}. \tag{8}$$

while in modified Gram-Schmidt, it is computed as

$$\Pi^{(j)} = (I_{n\times n} - \mathbf{q}_j \mathbf{q}_j^{\mathrm{T}})(I_{n\times n} - \mathbf{q}_{j-1}\mathbf{q}_{j-1}^{\mathrm{T}}) \ldots (I_{n\times n} - \mathbf{q}_1 \mathbf{q}_1^{\mathrm{T}}). \tag{9}$$

For randomized variants, the standard Gram–Schmidt procedure operates on the sketched matrix ΘX. Since the columns of R are computed sequentially in each iteration, the corresponding columns of the Θ-orthonormal factor Q can likewise be formed incrementally within the same loop.

To describe the standard and randomized Gram–Schmidt processes within a unified framework, we introduce an oblique projector [13]. Suppose $X = QR$, where the columns of Q form a Θ-orthonormal basis of the subspace $\mathcal{X} \subset \mathbb{R}^n$, i.e., $(\Theta Q)^{\mathsf{T}}(\Theta Q) = I$. We define the operator

$$P_{\mathcal{X}} := Q(\Theta Q)^{\mathsf{T}}\Theta.$$

Since $P_{\mathcal{X}}$ is idempotent but not self-adjoint, it follows that $P_{\mathcal{X}}$ is an oblique projection onto the subspace $\mathcal{X} \subset \mathbb{R}^n$.

Instead of the orthogonal projection operator Π employed in the standard Gram–Schmidt procedure, as shown in Equation (8) and Equation (9), the randomized Gram–Schmidt method relies on an oblique projection, referred to as the *sketch-orthogonal projector*, denoted by Ω.

For rCGS, at the j-th iteration, this operator is given by

$$\Omega^{(j)} = I_{n\times n} - P_{\mathcal{V}_j}, \qquad P_{\mathcal{V}_j} = Q_j(\Theta Q_j)^{\mathsf{T}}\Theta\,,$$

and is applied to the vector $\mathbf{x}_{j+1}$ to remove its component in the previously constructed subspace $\mathcal{V}_j$.

In rMGS, this operator takes the form

$$\Omega^{(j)} = (I_{n\times n} - P_{q_j})(I_{n\times n} - P_{q_{j-1}}) \ldots (I_{n\times n} - P_{q_1})\,, \qquad P_{q_j} = q_j(\Theta q_j)^{\mathsf{T}}\Theta\,.$$

The resulting randomized Gram–Schmidt procedure is summarized in Algorithm 6.

Algorithm 6: Randomized standard Gram-Schmidt

Input: Matrix $X \in \mathbb{R}^{n\times m}$ with $m \le n$. Random sketch $\Theta \in \mathbb{R}^{d\times n}$, $m \le d \ll n$.
Output: $n \times m$ Θ-orthonormal matrix Q and $m \times m$ upper triangular factor R such that $X = QR$.
for $i = 1$ **to** m **do**
  Compute $\mathbf{q}_i' = \Omega^{(i-1)}\mathbf{x}_i$, also yielding $R(1 : i-1, i)$;
  $R(i,i) = \|\Theta\mathbf{q}_i'\|$;
  $\mathbf{q}_i = \mathbf{q}_i' \,/\, R(i,i)$;
end
return $Q = [\mathbf{q}_1, \ldots, \mathbf{q}_m]$ *and* R

4.2.2 Randomized Gram-Schmidt algorithm

The randomized Gram–Schmidt (RGS) process [5; 18] is derived from the *sketch-and-solve* paradigm in randomized numerical linear algebra. The central idea of RGS is to compute each column of the upper triangular factor R by solving a sketched least-squares problem of reduced dimension,

$$R(1 : i-1, i) = \arg\min_{\mathbf{y}} \|S_{i-1}\mathbf{y} - \mathbf{p}_i\|, \tag{10}$$

where $S_{i-1} = \Theta Q_{i-1}$ denotes the sketched orthonormal basis obtained from the previously computed columns, and $\mathbf{p}_i = \Theta \mathbf{x}_i$ is the sketched version of the current vector to be orthogonalized.

Compared with the classical Gram–Schmidt procedure, which requires computing inner products and projections in the original n-dimensional space, RGS replaces these operations by a least-squares problem of size $d \times (i-1)$, with $d \ll n$. As a result, the cost of computing each column of R is significantly reduced, leading to a lower overall computational complexity, especially when n is large.

Moreover, under suitable subspace embedding conditions on the sketching matrix Θ, the solution of the sketched least-squares problem provides an accurate approximation to the solution of the original least-squares problem. This ensures that the resulting factor R faithfully captures the projection coefficients up to controlled distortion, while substantially reducing computational cost. The complete RGS procedure is summarized in Algorithm 7.

Algorithm 7: Randomized Gram-Schmidt (Algorithm 2 in [5])

Input: Matrix $X \in \mathbb{R}^{n\times m}$ with $m \le n$. Random sketch $\Theta \in \mathbb{R}^{d\times n}$, $m \le d \ll n$.
Output: $n \times m$ Θ-orthonormal matrix Q and $m \times m$ upper triangular factor R such that $X = QR$.
for $i = 1 : m$ **do**
  Sketch $\mathbf{x}_i$: $\mathbf{p}_i = \Theta \mathbf{x}_i$;
  Solve $k \times (i-1)$ least-squares problem: $R(1 : i-1, i) = \arg\min_{\mathbf{y}} \|S_{i-1}\mathbf{y} - \mathbf{p}_i\|$;
  Compute projection of $\mathbf{x}_i$: $\mathbf{x}'_i = \mathbf{x}_i - Q(:, i-1)R(1 : i-1, i)$;
  Sketch $\mathbf{q}'_i$: $\mathbf{s}'_i = \Theta \mathbf{q}'_i$;
  Compute the sketched norm $r_{ii} = \|\mathbf{s}'_i\|$, $R(i, i) = r_{ii}$;
  Scale vector $\mathbf{s}_i = \mathbf{s}'_i / r_{ii}$;
  Scale vector $\mathbf{q}_i = \mathbf{q}'_i / r_{ii}$;
end
return $Q = [\mathbf{q}_1, \ldots, \mathbf{q}_m]$ *and* R

4.2.3 Randomized Cholesky QR factorization

In exact arithmetic, the Cholesky factor of $X^\mathsf{T}X$ coincides with the R factor in the QR factorization of X, since

$$X^\mathsf{T}X = R^\mathsf{T}Q^\mathsf{T}QR = R^\mathsf{T}R,$$

where Q has orthonormal columns. This observation underlies the close connection between QR factorization and Cholesky-based orthogonalization methods.

In the randomized setting, this idea is extended by replacing $X^\mathsf{T}X$ with a sketched approximation. Specifically, we compute the upper triangular factor of the smaller matrix ΘX (which is also the Cholesky factor of $(\Theta X)^\mathsf{T}(\Theta X)$) and use it as an approximate Cholesky factor for $X^\mathsf{T}X$. When the sketching matrix Θ satisfies an appropriate subspace embedding property, the matrix $(\Theta X)^\mathsf{T}(\Theta X)$ provides a spectrally accurate approximation of $X^\mathsf{T}X$, ensuring that the resulting factor reliably captures the geometry of the column space of X.

This randomized Cholesky approach offers two key advantages. First, it significantly reduces computational cost, as forming and factorizing the sketched matrix involves only $d \times n$ operations with $d \ll n$, instead of working directly with the full $n \times n$ normal matrix. Second, by avoiding the explicit formation of $X^\mathsf{T}X$, which is well known to amplify conditioning issues, the randomized approach often exhibits improved numerical stability in finite-precision arithmetic. These properties make randomized Cholesky factorization particularly attractive in large-scale and communication-avoiding settings.

We summarize the Randomized Cholesky QR in Algorithm 8 [4].

Algorithm 8: rCholesky (Algorithm 2 in [4])

Input: Matrix $X \in \mathbb{R}^{n\times m}$ with $m \le n$. Random sketch $\Theta \in \mathbb{R}^{d\times n}$, $m \le d \ll n$.
Output: $n \times m$ Θ-orthonormal matrix Q and $m \times m$ upper triangular factor R such that $X = QR$.
Compute $P = \Theta X$;
Compute $R = qr(P)$;
$Q = XR^{-1}$;
return $Q = [\mathbf{q}_1, \ldots, \mathbf{q}_m]$ *and* R

4.3 A Brief Analysis of the Accuracy and Stability of Randomized Orthogonalization Methods

In this subsection, we provide a qualitative analysis of the stability and reliability of the proposed randomized orthogonalization methods. More detailed and technical error analyses can be found in [4; 5; 9].

For the randomized orthogonalization schemes rCGS, rMGS, and rCGS2, the approximate orthogonality of the resulting Θ-orthonormal factor Q is guaranteed by the oblivious subspace embedding (OSE) property in Definition 2.2, which ensures approximate preservation of inner products under the sketching operator Θ. Consequently, the Gram matrix formed in the sketched space provides a reliable approximation of the exact Gram matrix, also leading to a stable computation of the upper-triangular factor R.

In the case of RGS, the accuracy of R follows from the sketch-and-solve analysis presented in Lemma 2.1, which shows that solving the sketched normal equations yields, with high probability, an approximation that is sufficiently close to the exact solution.

Although these randomized orthogonalization schemes differ in their algorithmic structures, they are all equivalent in exact arithmetic. This equivalence is made precise in Proposition 2.2, which establishes that the approximate inner-product preservation condition in Definition 2.2 and the sketch-and-solve accuracy condition in Lemma 2.1 are mathematically equivalent. As a consequence, all the randomized orthogonalization methods considered here compute a Θ-orthonormal factor Q and an upper-triangular factor R that is stable in expectation and with high probability, provided that the sketch dimension is chosen sufficiently large.

It is important to emphasize that, although these methods are equivalent in exact arithmetic, they accumulate rounding errors in different ways when implemented in floating-point arithmetic. As a result, they may exhibit distinct numerical behavior and varying sensitivity to such errors when embedded within the s-step GMRES framework.

5 Randomized s-step GMRES

Now we integrate the above RBGS method into the GMRES to solve a linear system.

5.1 Basis generation

To generate the s basis vectors at each step while reducing communication cost and enabling applicability to large-scale parallel computation, we avoid relying on the Arnoldi algorithm, which orthogonalizes each new vector against all previously generated vectors at every step. Instead, we generate s vectors simultaneously at each step and process them using *intra-block orthogonalization* followed by *inter-block orthogonalization*.

The effectiveness of this block-based strategy, however, critically depends on the quality of the basis vectors generated within each block. In finite-precision arithmetic, poorly chosen basis vectors may become nearly linearly dependent, leading to rapid loss of orthogonality during both intra- and inter-block orthogonalization. Such orthogonality loss can severely degrade numerical stability and, in turn, adversely affect the convergence behavior and the accuracy of the computed solution. Therefore, the choice of a suitable basis generation mechanism plays a central role in the overall robustness of s-step Krylov subspace methods.

A possible way to generate the basis for a Krylov subspace is using the matrix power kernel (MPK) [16]. It requires a matrix $A \in \mathbb{R}^{n\times n}$, a start vector $\mathbf{v}_1 \in \mathbb{R}^n$, and a sequence of polynomials $\{p_j(A)\}_{j=0}^{s}$ defined by the recurrence relation

$$p_{i+1}(A) = (A - \beta_i I)p_i(A)/\gamma_i$$

for $i = 1, \ldots, s$ and $\gamma_i \neq 0$. Thus the new Krylov basis vectors are given by

$$V(:,\ 2 : s+1) = [\mathbf{v}_2, \ldots, \mathbf{v}_{s+1}] = \left[p_1(A)\mathbf{v}_1, \ldots, p_s(A)\mathbf{v}_1\right].$$

We can also deduce that the corresponding change-of-basis matrix $B \in \mathbb{R}^{(s+1)\times s}$ is of the following form:

$$\begin{bmatrix} \beta_1 & & & \\ \gamma_1 & \beta_2 & & \\ & \gamma_2 & \ddots & \\ & & \ddots & \beta_s \\ & & & \gamma_s \end{bmatrix}$$

such that

$$AV(:,\ 1 : s) = V(:,\ 1 : s+1)B\,. \tag{11}$$

When setting $\gamma_i = 1$ and $\beta_i = 0$, we have the most fundamental basis, which is called the monomial basis:

$$V_{monomial} = \left[\mathbf{v}_1, A\mathbf{v}_1, A^2\mathbf{v}_1, \ldots, A^s\mathbf{v}_1\right].$$

Since the monomial basis can become ill-conditioned quickly, which limits the step size to very small s, for example, $s = 5$ is a common choice in practice [16; 30]. However, in practice it is often desirable to take a larger step size s in order to reduce communication. For the purpose of improving the numerical stability and enable larger values of s in the MPK, the Newton basis can be employed as a good choice. The Newton basis uses Ritz values, which approximates the eigenvalues of $\mathbf{A}$, as shifts to differ from the monomial basis as much as possible [16]. The Ritz values are computed by performing s iterations of the Arnoldi method. The coefficients in MPK corresponds to the Newton basis are $\beta_i = \theta_i$ and $\gamma_i = 1$ for all i, resulting in

$$V_{Newton} = \left[\mathbf{v}_1, (A - \theta_1 I)\mathbf{v}_1, \ldots, \prod_{i=1}^{s}(A - \theta_i I)\mathbf{v}_1\right],$$

where $\theta_1, \theta_2, \ldots, \theta_s$ are the Ritz values of A.

5.2 RBGS-GMRES

The standard deterministic GMRES can be viewed as the 1-step GMRES method, in which a single new Krylov basis vector is generated and orthogonalized at each iteration. In contrast, deterministic s-step GMRES constructs and processes *blocks* of Krylov basis vectors in each outer iteration. Rather than incrementally extending the Krylov basis via repeated applications of the Arnoldi process (see Algorithm 1), the method generates a block of s candidate basis vectors simultaneously, for example, using MPK:

$$\left[p_1(A)\mathbf{v}_1, \ldots, p_s(A)\mathbf{v}_1\right],$$

where $\{p_i\}_{i=1}^{s}$ are suitably chosen polynomials. In exact arithmetic, this polynomial-based construction spans the same Krylov subspace as classical GMRES, while offering greater flexibility in basis selection and improved communication efficiency in finite-precision. However, in finite-precision arithmetic, the choice of $\{p_i\}_{i=1}^{s}$ must be made carefully and depends on the specific problem.

To ensure numerical stability and maintain orthogonality, these vectors are subsequently orthogonalized using a *two-level orthogonalization scheme*: an inter-block orthogonalization against previously computed blocks, followed by an intra-block orthogonalization to enforce mutual orthogonality within the current block. This procedure is typically realized via a block Gram–Schmidt (BGS) process, as shown in Algorithm 2. By aggregating multiple matrix–vector products and orthogonalization operations, deterministic s-step GMRES substantially reduces global synchronization and communication costs relative to the classical Arnoldi-based implementation, making it particularly attractive for large-scale parallel environments and modern computational clusters.

The RBGS-GMRES algorithm is a randomized variant of the s-step GMRES method. It employs the randomized intra-block and inter-block orthogonalization strategies to construct a sketch-orthonormal basis for the Krylov subspace as shown in Algorithm 3.

To solve the linear system $A\mathbf{x} = \mathbf{b}$ with RBGS-GMRES, we can set the initial guess as $\mathbf{0}$, with the start vector $\mathbf{b}$, we obtain $r_{1,1} = \|\Theta\mathbf{b}\|$. The extra steps in RBGS-GMRES involve computing the upper Hessenberg matrix H, solving the small least-square $\mathbf{y} = \arg\min_{\mathbf{z}}\|H\mathbf{z} - r_{1,1}\mathbf{e}_1\|$ and computing an approximate solution $\mathbf{x} \approx Q(:,\ 1:k)\,\mathbf{y}$.

When computing the Hessenberg matrix H, the sketching technique accelerates such procedure by reducing the associated least-squares problem to a sketched least-squares problem of smaller dimension:

$$H = \arg\min_{Y}\|SY - \Theta AQ\| \tag{12}$$

instead of

$$H = \arg\min_{Y}\|QY - AQ\| \tag{13}$$

by the property of the subspace embedding. In the numerical experiment, solving the linear system Equation (12) also shows more stability than computing $H = (\Theta Q)^T(\Theta AQ)$.

RBGS-GMRES also minimizes the sketched residual norm instead of Equation (2). This becomes evident by expressing the least-squares problem as follows:

$$\begin{aligned}\arg\min_{\mathbf{z}}\|H\mathbf{z} - r_{1,1}\mathbf{e}_1\| &= \arg\min_{\mathbf{z}}\|\Theta Q(H\mathbf{z} - r_{1,1}\mathbf{e}_1)\| \\ &= \arg\min_{\mathbf{z}}\|\Theta(\mathbf{b} - QHz)\| \\ &= \arg\min_{\mathbf{z}}\|\Theta(\mathbf{b} - AQ(:, 1:m-1)z)\|\ .\end{aligned}$$

By Lemma 2.1, we deduce that the sketched residual norm of RBGS-GMRES differs from the residual norm by a factor $\sqrt{(1+\epsilon)/(1-\epsilon)}$. More specifically, let $\mathbf{x}^*$ be the approximate solution of standard GMRES and let $\tilde{\mathbf{x}}$ be the approximate solution of GMRES. The residual norm with respect to $\tilde{\mathbf{x}}$ is bounded by

$$\|\mathbf{b} - A\mathbf{x}^*\| \le \|\mathbf{b} - A\tilde{\mathbf{x}}\| \le \left(\frac{1+\epsilon}{1-\epsilon}\right)^{1/2}\|\mathbf{b} - A\mathbf{x}^*\|\ .$$

The complete implementation of RBGS-GMRES is summarized in Algorithm 9.

Algorithm 9: RBGS_GMRES

Input: Matrix $A \in \mathbb{R}^{n\times n}$ with $m \le n$. Right hand side $\mathbf{b} \in \mathbb{R}^n$. Step size s. Dimension of Krylov subspace $m = sp + 1$. Random sketch $\Theta \in \mathbb{R}^{d\times n}$, $m < d \ll n$. A Krylov subspace basis generation function `basisFunc`. Orthogonalization method for inter-block. Orthogonalization method for intra-block.

Output: Approximate solution $\mathbf{x} \in \mathbb{R}^n$. Relative residual at each iteration *relErr*. Orthogonaliztion loss of sketched matrix at each iteration *orthErr*.

$r_{1,1} = \|\Theta\mathbf{b}\|$;
$V(:,1) = \mathbf{b}$;
$R(1,1) = r_{1,1}$;
$Q(:,1) = V(:,1)/R(1,1)$;
$S(:,1) = \Theta Q(:,1)$;
for $j = 1$ **to** p **do**
  $i = s(j-1)+1$;
  $cols = i+1 : i+s$;
  $k = js$;
  $V(:,\ cols) =$ `basisFunc`$(A, Q(:,\ i), s)$;
  Θ-orthogonalize $V(:,\ cols)$ **with** $Q(:,\ 1:i)$**:**
  $[Q(:,\ cols), R(1:i,\ cols)] = Algorithm\ 5\,(Q(:,\ 1:i),\ S(:,1:i),\ V(:,\ cols), \Theta)$;
  Θ-orthonormal $Q(:,\ cols)$**:**
  $[Q(:,\ cols), R(cols,\ cols)] = Algorithm\ 4\,(Q(:,\ cols), \Theta)$;
  $S(:,\ cols) = \Theta Q(:,\ cols)$;
  Compute the orthogonalization loss for sketched matrix: $orthErr = \left[orthErr; \|S^{\mathbf{T}}_{1:i+s} S_{1:i+s} - I\|\right]$;
  $b_0 = i : k$;
  Update the Hessenberg matrix: $H(1:i+s,\ b_0) = \arg\min_Y \|S(:,\ 1:i+s)\,Y - \Theta A Q(:,\ b_0)\|$;
  Solve the least square problem: $\mathbf{y} = \arg\min_{\mathbf{z}} \|H\mathbf{z} - r_{1,1}\mathbf{e}_1\|$;
  Compute the relative residual: $relRes = [relRes;\ \|AQ(:,\ 1:k)\mathbf{y} - \mathbf{b}\|/\|\mathbf{b}\|]$;
end
Add correction from current Krylov subspace: $\mathbf{x} = \mathbf{x}_0 + Q(:,\ 1:k)\,\mathbf{y}$;

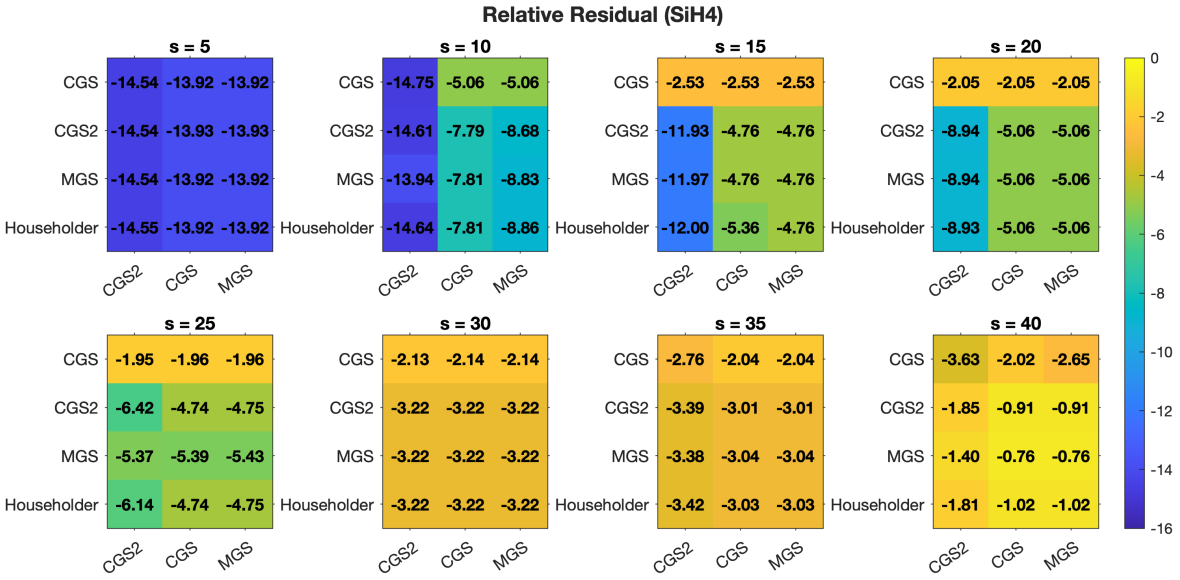


Figure 1: Relative residual heatmap for the SiH4 using the BGS method.

6 Numerical experiments

This section compares the numerical performance of deterministic and randomized s-step GMRES methods under various combinations of intra-block and inter-block orthogonalization strategies. The information of the test matrices are showed in Table 1. Among these, the matrices "SiH4", "Si10H16" and "SiO2" are from the SuiteSparse matrix collection, the matrix "genMatrix" is generated by a unitary similarity transformation $A = F^*BF$, where F is the discrete Fourier transform (DFT) matrix and B is a diagonal matrix with positive diagonal entries logarithmically spaced between 10^{-3} and 1. Here, `n` denotes the size of the test matrices. The parameter `ctol` represents the convergence tolerance used as the stopping criterion for the iterative method and is set to a very small value, 10^{-16}, in order to evaluate the long-term performance of each orthogonalization method. The parameter `m` denotes the dimension of the Krylov subspace and is selected according to the size of the test matrices.

Taking both computational efficiency and numerical stability into account, we employ **sparse sign** as the subspace embedding with the sparsity parameter $\xi = 8$ throughout all subsequent numerical experiments.

matrix	n	condition number	ctol	m
SiH4	5 041	1.065536e+03	10^{-16}	500
genMatrix(1e+3)	5 000	1e+03	10^{-16}	1000
Si10H16	17 077	5.633416e+04	10^{-16}	1000
SiO2	155 331		10^{-16}	3000

Table 1: Test matrices

6.1 Effects of orthogonalization methods and step size on BGS-GMRES

Figure 1–Figure 6 present heatmaps of the relative residual, loss of orthogonality, and the corresponding iteration counts for the SiH4 matrix using BGS and RBGS under the Newton basis. In each figure, rows correspond to **intra-block** orthogonalization methods, while columns correspond to **inter-block** orthogonalization methods. For each method, the reported iteration count **k** in Figure 3 and Figure 6 corresponds to the earliest iteration at which the relative residual satisfies

$$\texttt{relRes\_vec}(k) < 5 \cdot \min_j \texttt{relRes\_vec}(j),$$

where `relRes_vec` denotes the relative residual recorded at each iteration for each experiment.

Figure 1–Figure 3 present the corresponding results for BGS-GMRES. For the relative residual, we can observe that when s is relatively small (e.g., $s = 5, 10$), the inter-block orthogonalization plays a more important role than intra-block orthogonalization—a numerically stable inter-block orthogonalization method

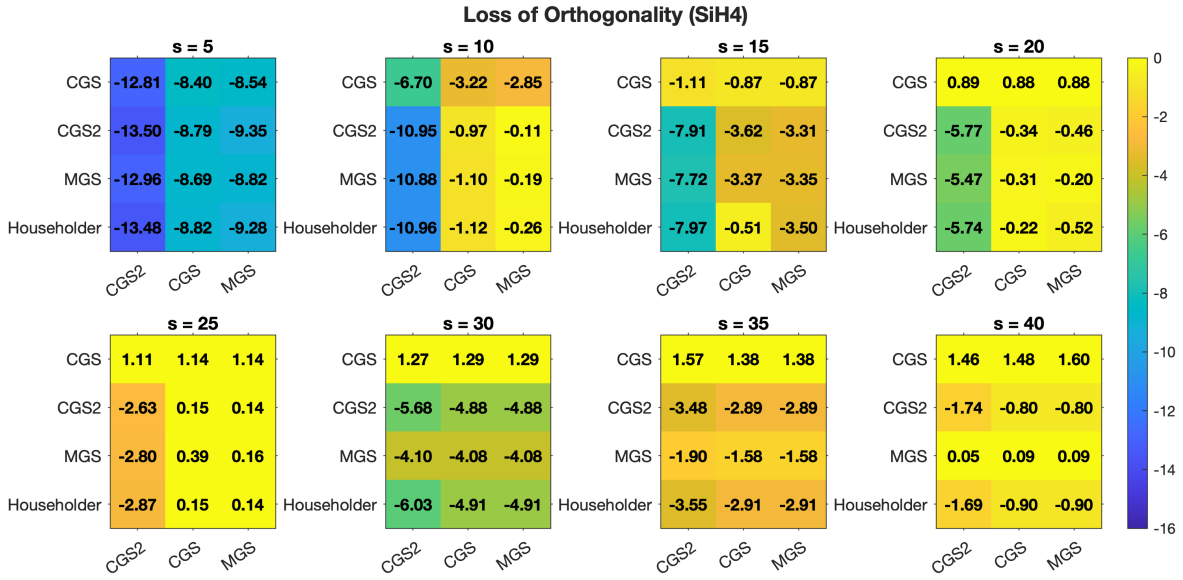


Figure 2: Loss of orthogonality heatmap for the SiH4 using the BGS method.

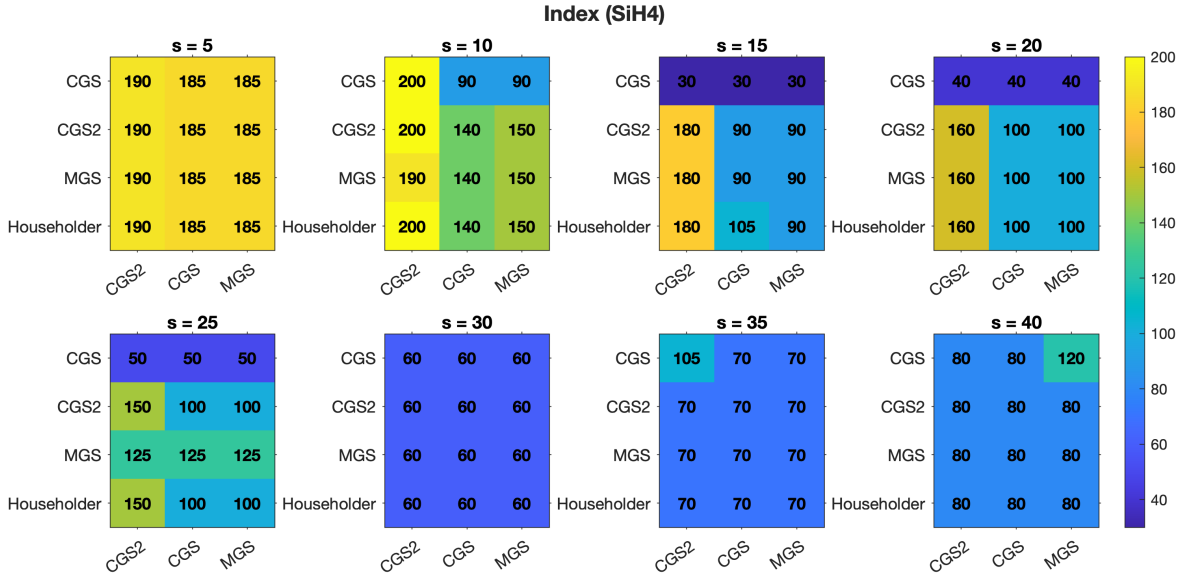


Figure 3: Iteration number heatmap for the SiH4 using the BGS method.

can still achieve a good relative residual even when combined with a relatively numerically unstable intra-block orthogonalization method; however, as s increases, we can observe that the intra-block method begins to dominate: as shown in Figure 1, when $s = 15, 20, 25$, the combination that uses CGS as an intra-block orthogonalization method achieve the relative residual that is noticeably worse than those of the other methods and corresponding show an early stagnation in Figure 3.

Another noteworthy observation is that the loss of orthogonality deteriorates earlier than the relative residual. By comparing Figure 2, Figure 1, and Figure 3, we observe that the loss of orthogonality consistently exhibits an earlier decay than the relative residual. For instance, when $s = 5$, a pronounced loss of orthogonality is already evident for CGS and MGS when used as inter-block orthogonalization methods, whereas the relative residuals and stagnation iteration counts remain comparable across all method combinations. Similarly, when $s = 10$, the CGS intra-block method displays a clear loss of orthogonality even when coupled with the relatively stable inter-block method CGS2; however, the noticeable difference in the relative residual is not observed until $s = 15$.

This behavior is consistent with the theoretical understanding of Krylov subspace methods under finite-precision arithmetic. In exact arithmetic, the Arnoldi process produces an orthonormal basis of the Krylov subspace, whereas in floating-point arithmetic the accumulation of roundoff errors inevitably leads to a gradual

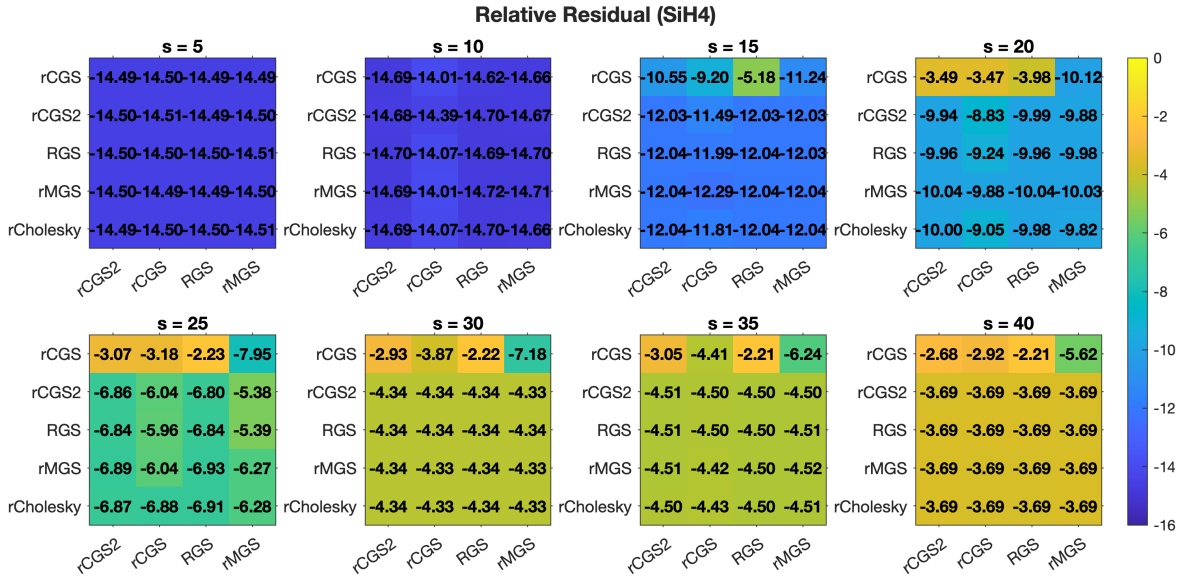


Figure 4: Relative residual heatmap for the SiH4 using the RBGS method.

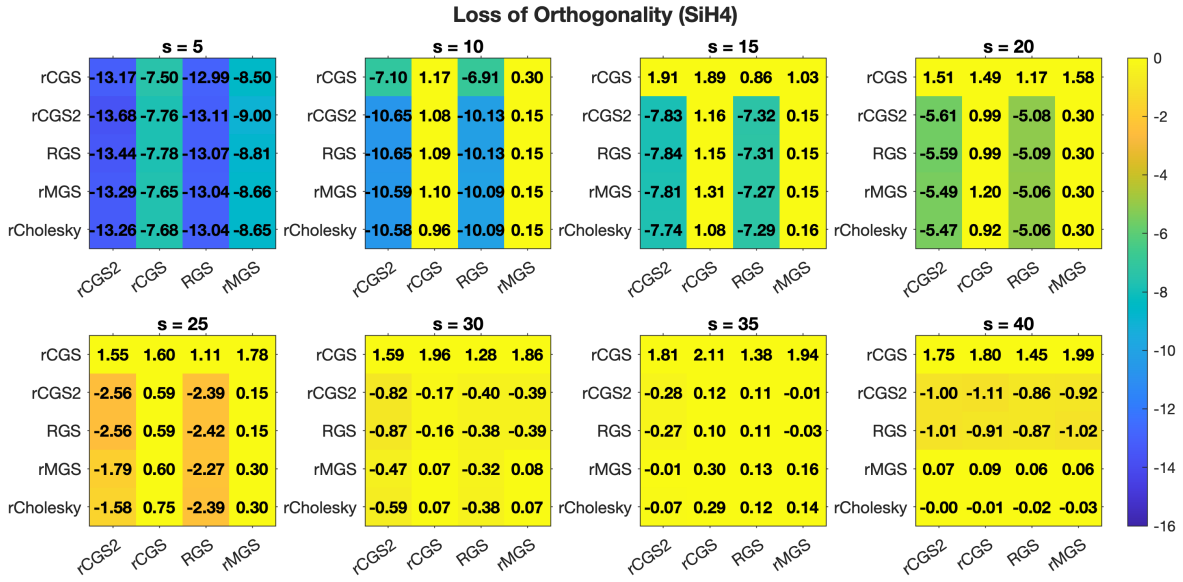


Figure 5: Loss of orthogonality heatmap for the SiH4 using the RBGS method.

loss of orthogonality. For linear system solvers such as GMRES, this loss of orthogonality is generally less critical than in eigenvalue computations, since phenomena such as the appearance of spurious or duplicated eigenvalues are not directly relevant. Nevertheless, the degradation of orthogonality reduces the effective dimension of the Krylov subspace and may impair the quality of newly generated search directions, thereby slowing down convergence. As a result, the loss of orthogonality often manifests itself earlier than a noticeable deterioration in the relative residual, which explains the delayed impact of orthogonality loss on the residual behavior observed in our experiments.

6.2 Comparison of BGS and RBGS in GMRES

With respect to the relative residual, Figure 1 and Figure 4 show that, when $s = 5$, both BGS and RBGS achieve relatively good accuracy across all orthogonalization method combinations. However, as s increases, BGS quickly loses accuracy in terms of the relative residual for certain less stable orthogonalization combinations. Additionally, RBGS helps mitigate early stagnation as s increases.

Furthermore, we observe that the dominance of the inter-block orthogonalization method is not evident in RBGS for relatively small step sizes s, as shown in Figure 4. Noticeable differences only emerge when rCGS

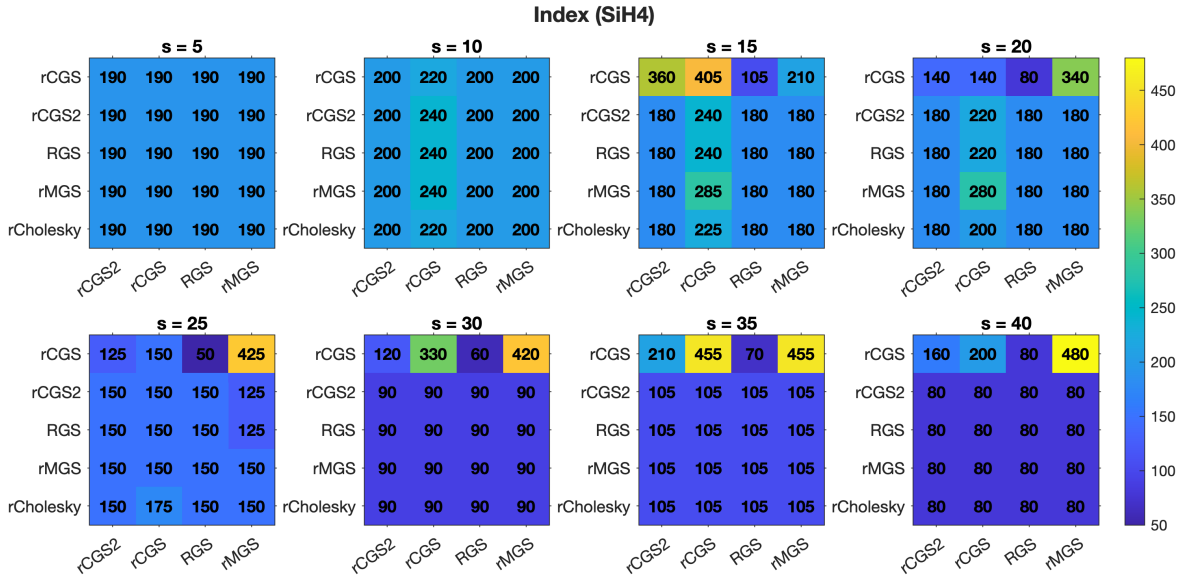


Figure 6: Iteration number heatmap for the SiH4 using the RBGS method.

is used as the intra-block orthogonalization method for larger values of s (e.g., $s = 15, 20, 25, 30, 35, 40$).

Figure 7 presents the entire iterative process for several orthogonalization schemes in both **BGS-GMRES** and **RBGS-GMRES**. It reports the relative residuals and the corresponding orthogonality losses for various combinations of orthogonalization strategies in BGS and RBGS, with a block size of $s = 15$, where the relative residual of the MATLAB built-in GMRES is also included as a benchmark. As can be observed, during the first few tens of iterations, all considered orthogonalization schemes closely overlap with the GMRES benchmark, indicating that the early convergence behavior is largely insensitive to the choice of orthogonalization strategy.

As the iteration count increases, however, a progressive loss of orthogonality becomes evident for all methods, with varying severity depending on the orthogonalization scheme. This degradation in orthogonality is accompanied by a clear stagnation or slowdown in the convergence of the relative residual, suggesting a strong correlation between orthogonality loss and the attainable accuracy of the Krylov basis.

In particular, schemes such as MGS/CGS and RGS/rCGS exhibit comparable levels of orthogonality loss throughout the iterations. Nevertheless, the randomized variants consistently achieve lower relative residuals than their deterministic counterparts, demonstrating improved robustness with respect to the accumulation of numerical errors. This indicates that randomized orthogonalization can mitigate the adverse effects of loss of orthogonality on convergence, even when the overall orthogonality levels are similar.

Moreover, we observe that methods suffering from more severe orthogonality loss tend to reach an accuracy plateau significantly earlier, whereas schemes with better orthogonality preservation continue to reduce the residual over a longer range of iterations. This highlights the critical role of orthogonality maintenance in long-term GMRES convergence.

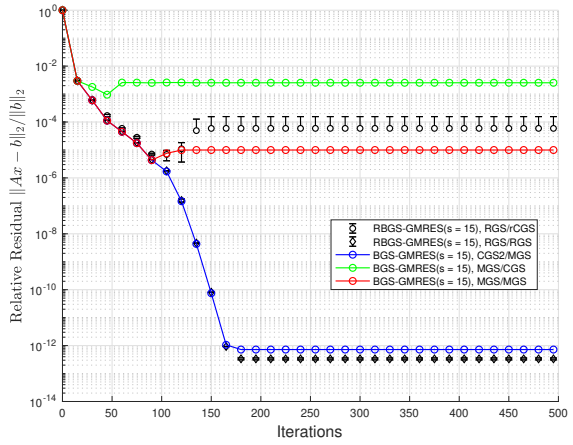
In addition, for $s = 15$, Figure 1 and Figure 4 show that BGS-GMRES attains comparable accuracy with most of the orthogonalization combination in RBGS-GMRES (except rCGS/rCGS2, rCGS/rCGS, rCGS/RGS) only under restrictive conditions. Specifically, comparable performance is observed in BGS-GMRES only when the most stable but computationally expensive inter-block orthogonalization method, CGS2, is employed and only in combination with sufficiently stable intra-block orthogonalization schemes. When less stable orthogonalization combinations are used, such as CGS/CGS2, the method fails to achieve comparable accuracy.

Furthermore, Figure 8 reports the minimum relative residual attained by RBGS-GMRES (using RGS/RGS) and BGS-GMRES (using CGS/CGS2) for step sizes s ranging from 5 to 50. For RBGS-GMRES, the minimum relative residual at each step size is obtained by selecting the minimum of the sample mean values of the relative residual over 20 independent runs.

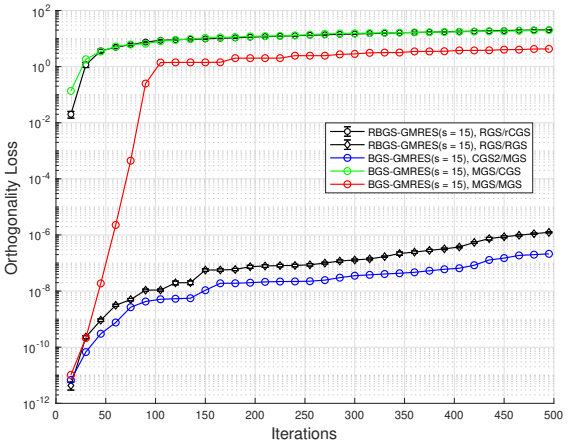
As shown in Figure 8, RBGS-GMRES consistently achieves smaller minimum relative residuals than BGS-GMRES across the entire range of step sizes, indicating that the randomized variant exhibits superior robustness with respect to the choice of block size. In particular, while both methods attain near machine precision for small values of s, the accuracy of BGS-GMRES deteriorates rapidly as s increases, whereas RBGS-GMRES maintains substantially lower residual levels over a wider range of step sizes.

These results further corroborate the observations from Figure 1–Figure 7, demonstrating that randomization effectively alleviates the adverse effects of orthogonality loss and error accumulation associated with

larger step sizes. In contrast, the deterministic BGS-GMRES becomes increasingly sensitive to the step size, even when relatively stable orthogonalization combinations are employed.



(a) Relative Residual [1]



(b) Orthogonality Loss [2]

Figure 7: Orthogonalization Schemes for SiH4 over 20 runs

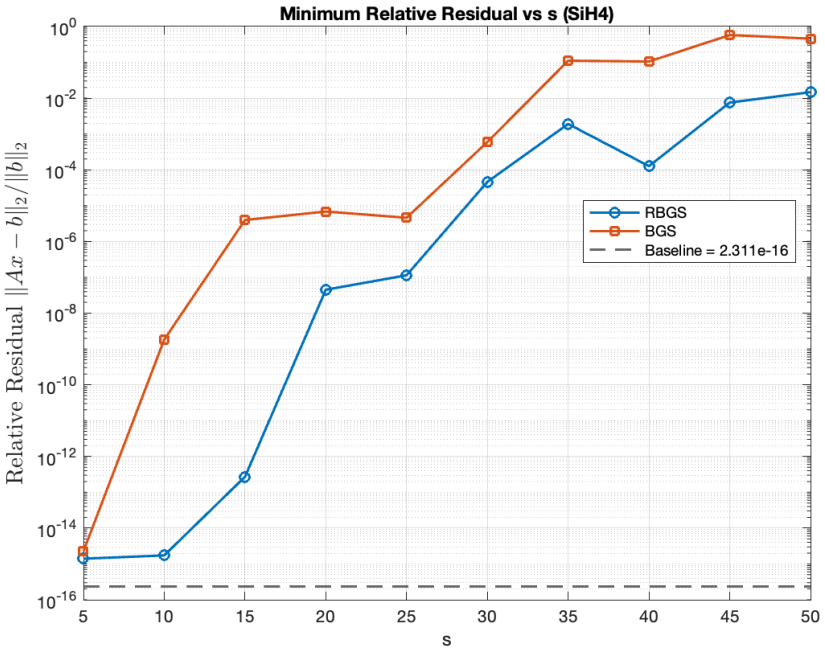


Figure 8: Minimum relative residual achieved by RBGS-GMRES (using RGS/RGS) and BGS-GMRES (using CGS/CGS2) for different step sizes s. The baseline corresponds to the minimum relative residual attained by the MATLAB built-in GMRES.

6.3 Comparison of different bases for RBGS-GMRES

We then compare the performance of RBGS-GMRES using the monomial and Newton bases on the same matrix SiH4 with an identical right-hand side $\mathbf{b}$. Figure 4– Figure 6 and Figure 9– Figure 11 present the heatmaps of the relative residual and the loss of orthogonality for SiH4 under the Newton and monomial bases, respectively, across different orthogonalization combinations and block sizes s.

[1] In the legend, the first orthogonalization method denotes the inter-block orthogonalization, while the second denotes the intra-block orthogonalization. For instance, RGS/rCGS indicates that RGS is used for inter-block orthogonalization and rCGS for intra-block orthogonalization. For the randomized methods, the error bars are constructed using the sample mean and the corresponding 5% confidence interval over 20 independent runs.

[2] Here, for deterministic method, the orthogonality loss is computed as $\|Q^{\mathsf{T}}Q - I\|_F$; for the randomized method, the orthogonality loss is computed as $\|(\Theta Q)^{\mathsf{T}}\Theta Q - I\|_F$.

Figure 9: Relative residual heatmap for the SiH4 using the RBGS method with monomial basis.

From Figure 4 and Figure 9, we observe that RBGS-GMRES exhibits qualitatively similar trends under both bases as the block size s increases. In particular, increasing s generally leads to a degradation in the achievable accuracy, accompanied by a pronounced loss of orthogonality, regardless of the orthogonalization scheme employed.

Nevertheless, notable quantitative differences arise between the two bases. For small block sizes, the monomial basis exhibits a significantly earlier stagnation in the relative residual compared with the Newton basis. This behavior can be attributed to the intrinsic convergence properties of the power method underlying the monomial basis, which tends to amplify numerical errors and accelerate the loss of orthogonality in finite-precision arithmetic.

The heatmaps further reveal a strong correlation between the loss of orthogonality and the attainable relative residual under both bases. Orthogonalization schemes that suffer from severe orthogonality loss consistently fail to achieve high accuracy, while methods that better preserve orthogonality remain effective over a wider range of block sizes. In this respect, the Newton basis demonstrates improved numerical robustness, delaying the onset of orthogonality degradation and residual stagnation when compared with the monomial basis.

Moreover, the results obtained with the monomial basis provide predictive insight into the behavior of RBGS-GMRES under the Newton basis for larger block sizes. Specifically, the early breakdown observed for the monomial basis at moderate values of s foreshadows a similar deterioration for the Newton basis as s becomes sufficiently large (e.g., $s > 40$), where all orthogonalization combinations eventually fail to deliver a satisfactory relative residual.

6.4 Delayed Convergence Phenomena of rMGS/rCGS in RBGS-GMRES

From Figure 4 and Figure 9, we observe that the orthogonalization scheme rMGS/rCGS (i.e., rMGS for inter-block orthogonalization and rCGS for intra-block orthogonalization) achieves the best relative residual among all orthogonalization combinations due to its delayed convergence for certain step sizes s, whereas the other schemes exhibit stagnation, as illustrated in Figure 12, Figure 13, Figure 14 and Figure 15 for 4 representative test matrices under the Newton bases when the step size $s = 30$.

In contrast to other orthogonalization combinations, which suffer from early stagnation of the relative residual, rMGS/rCGS continues to reduce the residual over a substantially longer range of iterations. As a result, rMGS/rCGS achieves the smallest relative residual among all considered schemes, despite not necessarily exhibiting the lowest orthogonality loss at early iterations. This observation indicates that the benefit of rMGS/rCGS does not stem from superior short-term orthogonality preservation, but rather from its ability to delay the onset of the severe orthogonality degradation that ultimately limits GMRES convergence.

The orthogonality loss plots further corroborate this interpretation. While all schemes experience a gradual accumulation of orthogonality errors, rMGS/rCGS maintains a more moderate growth rate over a prolonged iteration horizon. In contrast, schemes such as RGS/rCGS and RGS/rCGS2 exhibit a much more rapid deterio-

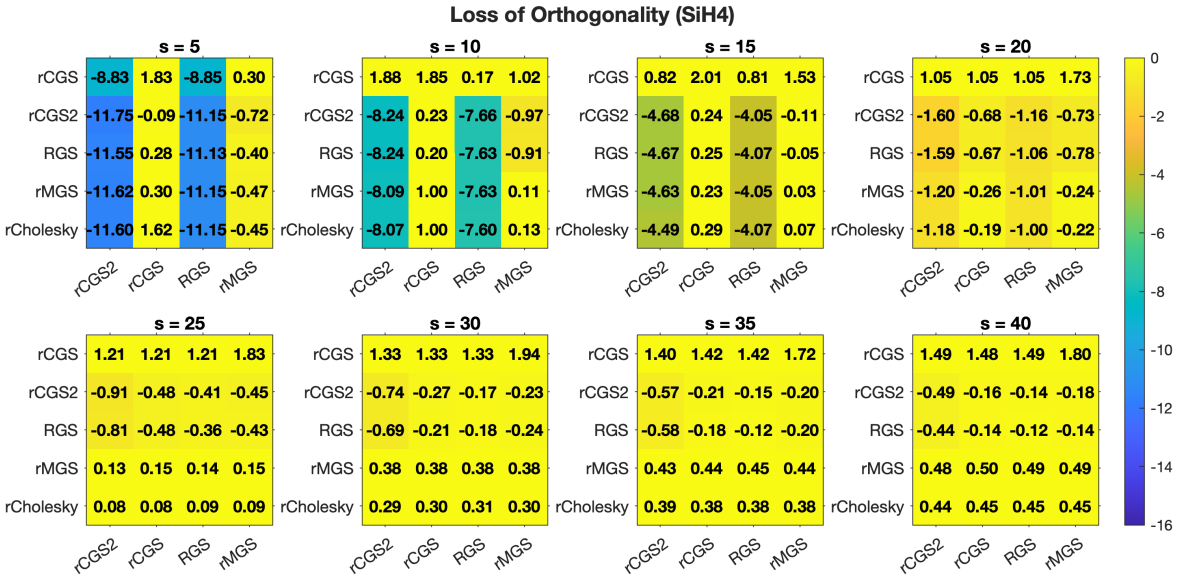


Figure 10: Loss of orthogonality heatmap for the SiH4 using the RBGS method with monomial basis.

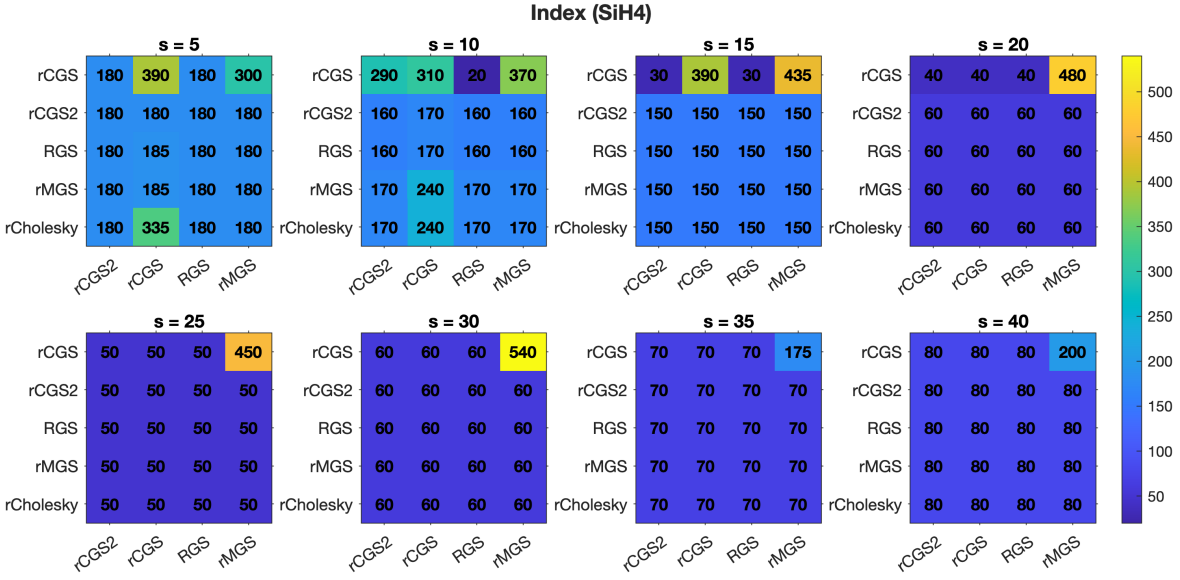
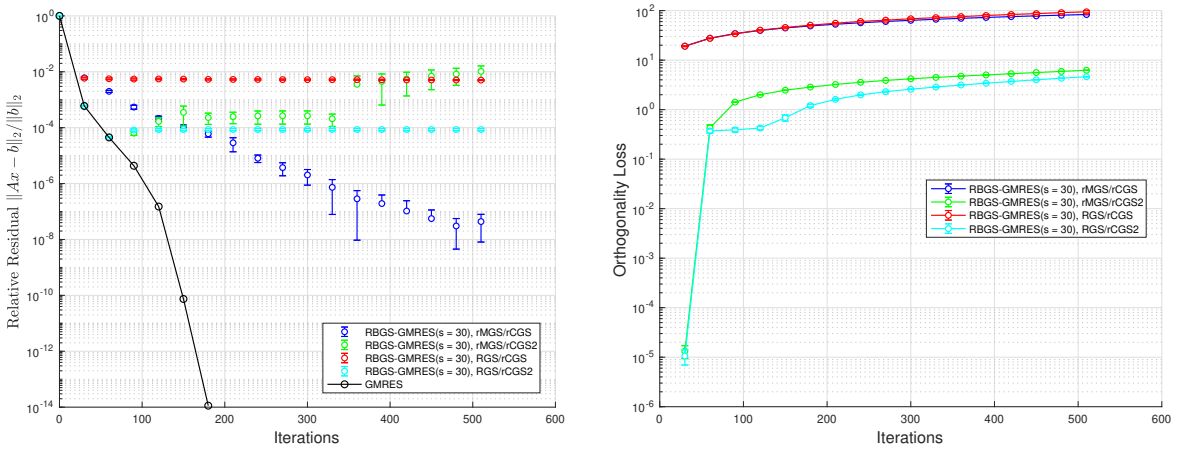


Figure 11: Iteration number heatmap for the SiH4 using the RBGS method with monomial basis.

ration, which directly correlates with the premature stagnation observed in their corresponding residual curves. Notably, although RGS/rCGS and rMGS/rCGS may exhibit comparable levels of overall orthogonality loss as measured by standard scalar metrics, their convergence behaviors differ markedly. This discrepancy suggests that global orthogonality loss alone does not fully characterize the quality of the generated Krylov subspace.

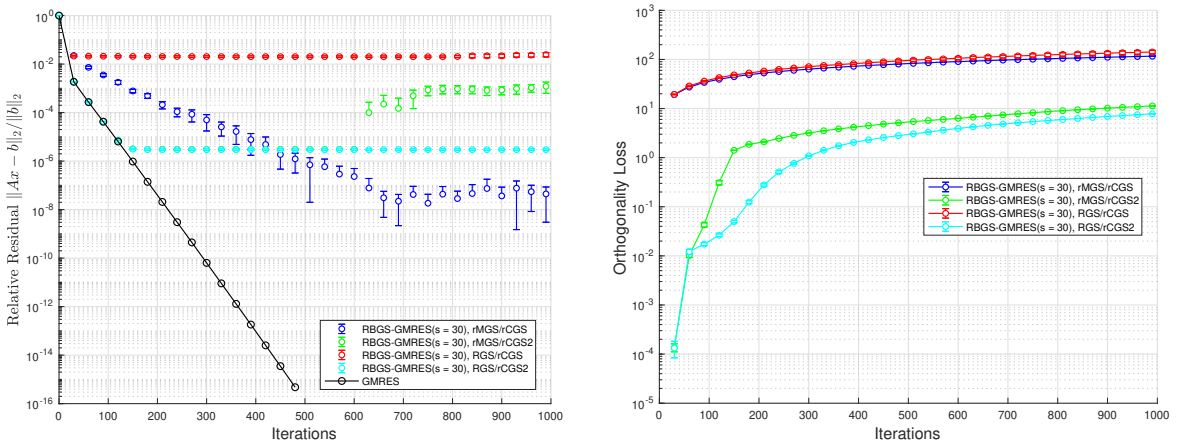
In particular, rMGS/rCGS appears to preserve critical search directions and the underlying Arnoldi structure more effectively, whereas RGS/rCGS suffers from a more detrimental distortion of the Krylov basis, leading to inferior residual reduction despite similar aggregate orthogonality measures. These results highlight that, within RBGS, the distribution and propagation of orthogonality errors play a more important role than their absolute magnitude.

Importantly, the delayed convergence behavior of rMGS/rCGS is consistently observed across all four test matrices, indicating that this phenomenon is intrinsic to the orthogonalization strategy itself rather than being problem-dependent. Overall, these findings suggest that employing a more stable inter-block orthogonalization can effectively postpone the numerical breakdown caused by loss of orthogonality, thereby extending the useful convergence phase of GMRES within the RBGS framework.



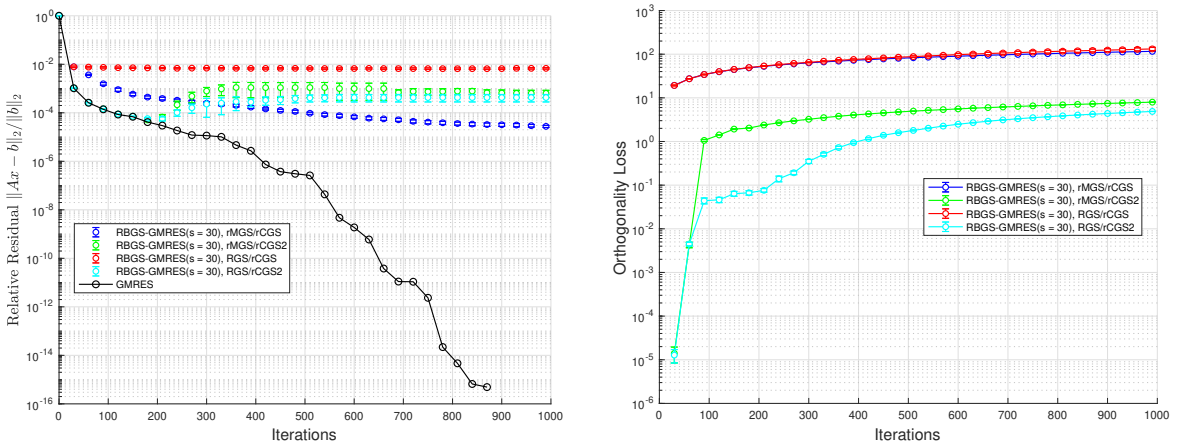
(a) Relative Residual (b) Orthogonality Loss

Figure 12: Different Randomized Orthogonalization Schemes for SiH4 over 20 runs



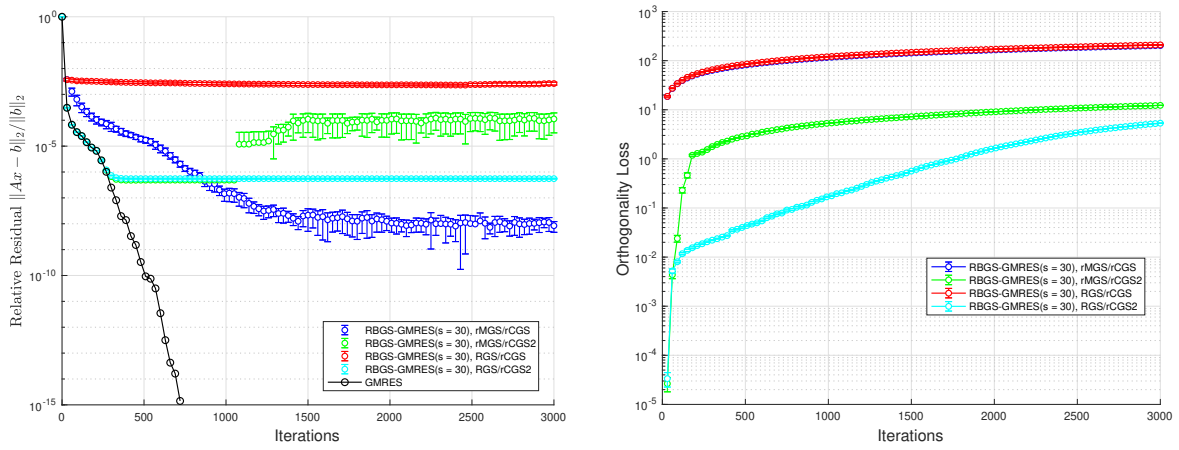
(a) Relative Residual (b) Orthogonality Loss

Figure 13: Different Randomized Orthogonalization Schemes for genMatrix(1e+3) over 20 runs



(a) Relative Residual (b) Orthogonality Loss

Figure 14: Different Randomized Orthogonalization Schemes for Si10H16 over 20 runs



(a) Relative Residual

(b) Orthogonality Loss

Figure 15: Different Randomized Orthogonalization Schemes for SiO2 over 20 runs

7 Conclusion

In this project, we extend the deterministic s-step GMRES framework to its randomized counterpart by incorporating various randomized orthogonalization strategies. Although these methods are mathematically equivalent in exact arithmetic, our study demonstrates that they accumulate rounding errors in fundamentally different ways when implemented in finite-precision arithmetic, leading to markedly different numerical behaviors.

Through extensive numerical experiments, we observe that randomized orthogonalization schemes often outperform their deterministic counterparts, even when paired with orthogonalization procedures that are traditionally considered less stable. In addition, the block size s plays a critical role in determining the performance of both deterministic and randomized s-step GMRES. For both deterministic and randomized s-step GMRES, different combinations of inter-block and intra-block orthogonalization lead to distinct accuracy and stability profiles, reflecting the diverse ways in which rounding errors are introduced and propagated throughout the iterations.

A key finding of this work is the delayed convergence phenomenon exhibited by certain randomized orthogonalization combinations, most notably rMGS/rCGS. While such schemes do not necessarily achieve superior orthogonality preservation in the early iterations, they are able to delay the onset of severe orthogonality degradation that ultimately limits GMRES convergence. This enables rMGS/rCGS to attain smaller relative residuals over a prolonged iteration horizon compared with other orthogonalization combinations that suffer from premature stagnation.

Importantly, we also perform a direct comparison between randomized and deterministic variants of GMRES within the s-step framework. The results demonstrate that randomization can significantly enhance the numerical stability of deterministic s-step GMRES. Notably, even when using computationally inexpensive orthogonalization schemes such as rCGS, the randomized methods are able to achieve accuracy comparable to, or in some cases exceeding, that of more computationally demanding deterministic schemes such as CGS2, provided that an appropriate block size s is chosen. These observations highlight that randomization fundamentally alters the way numerical errors are distributed and propagated, rather than simply reducing their overall magnitude.

Furthermore, our results emphasize the importance of basis generation in the design of robust s-step Krylov subspace methods. In particular, the choice of polynomial basis significantly influences numerical stability, as it governs the extent to which power-method-induced convergence and loss of linear independence are amplified. Employing more stable bases, such as the Newton basis, can substantially mitigate these effects and extend the effective convergence regime of the algorithm.

Overall, this work demonstrates that randomization provides a principled and effective mechanism for improving the accuracy, stability, and robustness of communication-avoiding GMRES methods. By carefully selecting the block size, orthogonalization strategy, and basis representation, one can achieve a favorable trade-off between numerical robustness and computational efficiency. These results indicate that randomized s-step GMRES is a particularly appealing approach for large-scale iterative solvers on modern HPC architectures, where communication efficiency, stability, and accuracy are all critical considerations.

As a possible direction for future work, it would be natural to examine the behavior of different orthogonalization combinations when embedded in other s-step Krylov subspace methods, such as s-step FOM. Since different orthogonalization combinations may lead to distinct patterns of rounding-error accumulation and propagation, they may influence convergence rates, the onset of stagnation, and orthogonality loss in FOM differently from their effects in GMRES: for example, orthogonalization schemes that perform well in s-step GMRES might exhibit different numerical behavior in s-step FOM. A systematic investigation of these effects could help clarify the role of orthogonalization in the design of stable and efficient s-step Krylov methods.

References

[1] N. Ailon and B. Chazelle, *Approximate nearest neighbors and the fast Johnson-Lindenstrauss transform*, in STOC'06: Proceedings of the 38th Annual ACM Symposium on Theory of Computing, ACM, New York, 2006, pp. 557–563.

[2] W. E. Arnoldi, *The principle of minimized iteration in the solution of the matrix eigenvalue problem*, Quart. Appl. Math., 9 (1951), pp. 17–29.

[3] Z. Bai, D. Hu, and L. Reichel, *A Newton basis GMRES implementation*, IMA J. Numer. Anal., 14 (1994), pp. 563–581.

[4] O. Balabanov, *Randomized cholesky qr factorizations*, 2022.

[5] O. Balabanov and L. Grigori, *Randomized Gram-Schmidt process with application to GMRES*, SIAM J. Sci. Comput., 44 (2022), pp. A1450–A1474.

[6] ——, *Randomized block Gram-Schmidt process for the solution of linear systems and eigenvalue problems*, SIAM J. Sci. Comput., 47 (2025), pp. A553–A585.

[7] O. Balabanov and A. Nouy, *Randomized linear algebra for model reduction. Part I: Galerkin methods and error estimation*, Adv. Comput. Math., 45 (2019), pp. 2969–3019.

[8] A. Björck, *Numerics of Gram-Schmidt orthogonalization*, vol. 197/198, 1994, pp. 297–316. Second Conference of the International Linear Algebra Society (ILAS) (Lisbon, 1992).

[9] E. Carson and Y. Ma, *On the backward stability of s-step GMRES*, SIAM J. Matrix Anal. Appl., 46 (2025), pp. 2008–2040.

[10] M. Charikar, K. Chen, and M. Farach-Colton, *Finding frequent items in data streams*, vol. 312, 2004, pp. 3–15. Automata, languages and programming.

[11] M. B. Cohen, *Nearly tight oblivious subspace embeddings by trace inequalities*, in Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms, ACM, New York, 2016, pp. 278–287.

[12] S. Dasgupta and A. Gupta, *An elementary proof of a theorem of Johnson and Lindenstrauss*, Random Structures Algorithms, 22 (2003), pp. 60–65.

[13] J.-G. de Damas, L. Grigori, I. Simunec, and E. Timsit, *Randomized orthogonalization and krylov subspace methods: principles and algorithms*, 2025.

[14] P. Drineas, M. W. Mahoney, and S. Muthukrishnan, *Sampling algorithms for l2 regression and applications*, in Proceedings of the Seventeenth Annual ACM-SIAM Symposium on Discrete Algorithm, SODA '06, USA, 2006, Society for Industrial and Applied Mathematics, p. 1127–1136.

[15] A. C. Gilbert, J. Y. Park, and M. B. Wakin, *Sketched svd: Recovering spectral features from compressive measurements*, 2012.

[16] M. Hoemmen, *Communication-avoiding Krylov subspace methods*, ProQuest LLC, Ann Arbor, MI, 2010. Thesis (Ph.D.)–University of California, Berkeley.

[17] W. B. Johnson and J. Lindenstrauss, *Extensions of Lipschitz mappings into a Hilbert space*, in Conference in modern analysis and probability (New Haven, Conn., 1982), vol. 26 of Contemp. Math., Amer. Math. Soc., Providence, RI, 1984, pp. 189–206.

[18] Y. Nakatsukasa and J. A. Tropp, *Fast and accurate randomized algorithms for linear systems and eigenvalue problems*, SIAM J. Matrix Anal. Appl., 45 (2024), pp. 1183–1214.

[19] J. Nelson and H. L. Nguy en, *OSNAP: faster numerical linear algebra algorithms via sparser subspace embeddings*, in 2013 IEEE 54th Annual Symposium on Foundations of Computer Science—FOCS 2013, IEEE Computer Soc., Los Alamitos, CA, 2013, pp. 117–126.

[20] V. Rokhlin and M. Tygert, *A fast randomized algorithm for overdetermined linear least-squares regression*, Proc. Natl. Acad. Sci. USA, 105 (2008), pp. 13212–13217.

[21] Y. Saad, *Iterative methods for sparse linear systems*, Society for Industrial and Applied Mathematics, Philadelphia, PA, second ed., 2003.

[22] Y. Saad and M. H. Schultz, *GMRES: a generalized minimal residual algorithm for solving nonsymmetric linear systems*, SIAM J. Sci. Statist. Comput., 7 (1986), pp. 856–869.

[23] T. Sarlos, *Improved approximation algorithms for large matrices via random projections*, in 2006 47th Annual IEEE Symposium on Foundations of Computer Science (FOCS'06), 2006, pp. 143–152.

[24] J. A. Tropp, A. Yurtsever, M. Udell, and V. Cevher, *Streaming low-rank matrix approximation with an application to scientific simulation*, SIAM J. Sci. Comput., 41 (2019), pp. A2430–A2463.

[25] R. Vershynin, *Introduction to the non-asymptotic analysis of random matrices*, 2011.

[26] M. J. Wainwright, *High-Dimensional Statistics: A Non-Asymptotic Viewpoint*, Cambridge Series in Statistical and Probabilistic Mathematics, Cambridge University Press, 2019.

[27] D. P. Woodruff, *Computational advertising: Techniques for targeting relevant ads*, Foundations and Trends® in Theoretical Computer Science, 10 (2014), p. 1–157.

[28] ——, *Sketching as a tool for numerical linear algebra*, Found. Trends Theor. Comput. Sci., 10 (2014), pp. iv+157.

[29] Z. Xu, J. J. Alonso, and E. Darve, *A numerically stable communication-avoiding s-step GMRES algorithm*, SIAM J. Matrix Anal. Appl., 45 (2024), pp. 2039–2074.

[30] I. Yamazaki, A. J. Higgins, E. G. Boman, and D. B. Szyld, *Two-stage block orthogonalization to improve performance of s-step gmres*, in 2024 IEEE International Parallel and Distributed Processing Symposium (IPDPS), 2024, pp. 26–37.