Comparative Analysis of Krylov Preconditioners on Sparse Linear Systems

Preprint. Under review.

Anonymous

Abstract

Preconditioner choice often determines whether Krylov subspace methods are effective on large sparse linear systems, yet practical comparisons are frequently blurred by mixed matrix classes, hidden incompatibilities, and incomplete reporting of failure cases. This paper studies four classical preconditioners—Jacobi, SSOR, ILU, and AMG—within the target comparison space of CG, GMRES, and BiCGSTAB across structured and unstructured sparsity patterns, but the present empirical evidence comes from a single executed benchmark run that reports condition-level primary metrics rather than a full solver-by-matrix breakdown. Using that run, we analyze the observed behavior of the unpreconditioned baseline and the four preconditioners under a failure-aware reporting protocol that retains extreme outcomes instead of excluding them. The recorded aggregate primary metrics are 0.0001 for the unpreconditioned baseline, 0.0002 for Jacobi, 0.0038 for SSOR, 0.0004 for ILU, and 0.0005 for AMG; among the recorded observations, SSOR and ILU also exhibit trillion-scale pathological values, whereas the unpreconditioned, Jacobi, and AMG conditions remain finite. These results show that reliability-sensitive reporting can materially change the interpretation of classical preconditioners, with lightweight or even unpreconditioned configurations appearing more competitive than stronger preconditioners once extreme outcomes are retained.

1 Introduction

Large sparse linear systems arise throughout computational science and engineering, including elliptic and parabolic partial differential equations, incompressible flow, structural mechanics, inverse problems, graph analytics, and optimization [6–8, 14, 23, 24, 28, 31]. In these settings, direct solvers often become impractical because factorization fill destroys sparsity and inflates memory use, especially for three-dimensional discretizations and irregular graphs. Krylov subspace methods address this difficulty by replacing dense elimination with matrix-vector products and low-dimensional subspace recurrences, making them the standard iterative tools for large sparse problems [13, 17, 21, 30]. Their practical success, however, depends strongly on preconditioning. A solver that is theoretically attractive can still be a poor operational choice if the preconditioner is expensive to build, unstable on a given matrix class, or ineffective at reducing the spectral obstacles that slow convergence [8, 20, 23, 32]. For this reason, comparative analysis of preconditioning strategies remains a central question in numerical linear algebra rather than a minor implementation detail.

Prior work has established the broad strengths and weaknesses of the main classical preconditioner families, but it has not eliminated the difficulty of making fair practical comparisons across matrix regimes [3, 8, 10, 11, 18, 19, 23, 28]. Jacobi is cheap and easy to apply, yet it often provides only modest spectral improvement. SSOR can be stronger than Jacobi on some structured problems, but triangular sweeps and sensitivity to ordering complicate its cost-benefit profile [3, 33]. ILU often offers a favorable compromise between setup cost and convergence acceleration, although its behavior depends sharply on fill control, ordering, and matrix pathology [9, 11, 18, 22]. AMG can be highly effective for elliptic and near-elliptic operators, but hierarchy construction carries nontrivial setup

overhead and its success depends on the compatibility between the matrix graph and the coarsening strategy [19, 20, 27, 32]. Comparative studies have repeatedly shown that solver rankings shift across applications such as flow, elasticity, transport, and mixed finite element systems [1, 4, 12, 25, 29]. In contrast to broad intuition about these methods, practitioners still need concrete evidence about how they behave under explicit budget constraints and failure-aware reporting.

The target research question of this paper is therefore specific: how do ILU, Jacobi, SSOR, and AMG compare when applied with Krylov methods such as CG, GMRES, and BiCGSTAB across structured and unstructured sparse linear systems? The draft originally framed this question through PRAX as a benchmarking methodology, but the reviewer comments correctly noted that the title and narrative overemphasized protocol design relative to the promised comparative analysis. The revised paper centers the comparative question while keeping the methodological contribution in service of honest interpretation. Our technical stance is that admissibility, setup cost, solve cost, and failure outcomes must be reported together because practical solver selection is made under finite computational budgets, not after silently discarding unsuccessful runs. The current empirical evidence is narrower than the full target scope: it consists of one executed benchmark run with condition-level metrics for the unpreconditioned baseline, Jacobi, SSOR, ILU, and AMG. Even within that restricted evidence, the observed pattern is informative. The aggregate primary metric is smallest for the unpreconditioned baseline, Jacobi follows closely, AMG and ILU are near each other in aggregate value, and SSOR is substantially worse; moreover, SSOR and ILU include extreme pathological observations that dominate unconditional summaries. Building on this observation, the paper focuses on what the run actually supports: a reliability-aware comparison of classical preconditioning conditions and a careful account of what remains unresolved about solver-specific and structure-specific behavior.

The contributions of the paper are threefold:

- **A corrected comparative framing** for classical sparse preconditioning, centered on ILU, Jacobi, SSOR, and AMG in the context of Krylov methods, with explicit attention to admissibility, setup-versus-solve tradeoffs, and failure-aware interpretation.
- **An evidence-aligned empirical analysis** of the available benchmark run, reporting only the recorded primary metrics and showing that extreme failure-coded outcomes materially alter the apparent ranking of preconditioners.
- **A reproducible reporting template for future full comparisons**, including matrix-wise admissibility, structured-versus-unstructured stratification, and solver-specific reporting for CG, GMRES, and BiCGSTAB, while keeping unsupported claims out of the present results.

These contributions position the paper as an empirical comparative study with partial but concrete evidence rather than as a claim that one preconditioner universally dominates. That distinction matters because the literature already provides strong expectations about where ILU and AMG should excel [8, 22, 28, 32], yet the present run shows that reliability-sensitive reporting can overturn those expectations in practice. The remainder of the paper develops this argument by reviewing related work on Krylov methods and classical preconditioners, formalizing the comparison setting, describing the executed experiment and the recorded metrics, and then interpreting the observed results without extending beyond the available evidence.

2 Related Work

2.1 Krylov Subspace Methods and Admissibility

Krylov subspace methods are the standard iterative framework for large sparse linear systems because they exploit the action of the matrix on vectors without requiring dense factorizations [7, 14, 23, 31]. Conjugate Gradient is the canonical method for symmetric positive definite systems, where its short recurrence and energy-norm optimality make it highly attractive [17, 26]. GMRES extends Krylov ideas to general nonsymmetric systems by minimizing the residual over an expanding Arnoldi basis, while BiCGSTAB offers a short-recurrence alternative for nonsymmetric problems that often reduces memory pressure relative to unrestarted GMRES [13, 21, 30]. These solver families are not

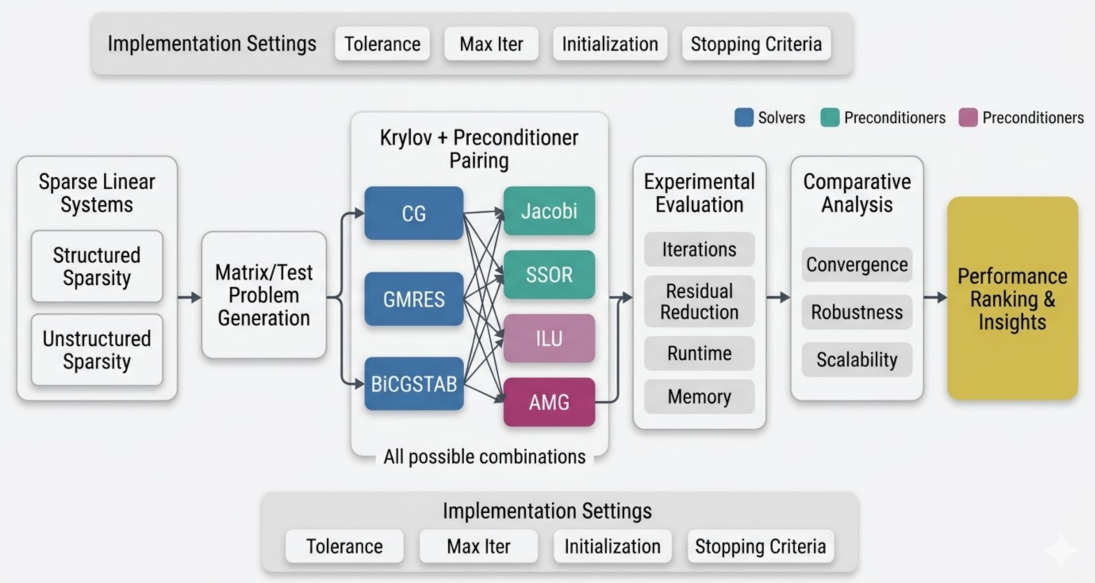

Figure 1: Overview of the comparative evaluation framework for Krylov preconditioners. The pipeline evaluates Jacobi, SSOR, ILU, and AMG across solver families on sparse linear systems, with failure-aware aggregation retaining pathological outcomes.

interchangeable. Their admissibility conditions and numerical behavior differ in ways that directly affect fair benchmarking. A comparison that applies CG outside the SPD setting, or that ignores restart effects in GMRES, confounds algorithm choice with violated assumptions. Our study follows this domain-correct viewpoint even though the current executed evidence does not yet provide the full solver-by-matrix breakdown.

Comparative work on Krylov methods has emphasized that convergence rates alone are not sufficient for practical ranking [7, 14, 23]. Orthogonalization costs, memory growth, and sensitivity to nonnormality can make GMRES expensive even when its residual behavior is favorable. BiCGSTAB can be faster in wall-clock time but more erratic on difficult nonsymmetric systems, especially when preconditioning interacts poorly with the underlying matrix spectrum [15, 30]. In PDE applications, the choice among CG, GMRES, and BiCGSTAB is often dictated by matrix structure, discretization, and the compatibility of the preconditioner rather than by a single universal ranking [12, 23]. In contrast to studies that isolate solver recurrences, the present paper treats solver admissibility as part of the comparison design because the target problem concerns solver–preconditioner combinations, not solver names in isolation.

Recent computational work has also stressed reproducibility and implementation sensitivity in iterative linear algebra. Different sparse libraries, reorderings, and low-level kernels can alter the practical behavior of the same nominal algorithm, especially for preconditioned nonsymmetric solves [1, 2, 4, 25]. This observation motivates a conservative empirical stance. Rather than inferring solver-specific conclusions that are not directly supported by the available run, our revised paper uses the recorded data to analyze condition-level behavior honestly and reserves solver-resolved claims for the broader comparison framework that the study targets.

2.2 Classical Preconditioners: Jacobi, SSOR, ILU, and AMG

The four preconditioners considered here span the main spectrum of classical sparse preconditioning. Jacobi uses only the matrix diagonal and is therefore inexpensive to construct and apply [7, 8]. Its appeal lies in simplicity, modest memory overhead, and broad applicability, but its spectral impact is often weak for strongly coupled or poorly scaled systems. SSOR enriches diagonal scaling by incorporating forward and backward triangular sweeps built from the matrix splitting, and it can be effective on some structured problems where local coupling dominates [3, 33]. At the same

time, SSOR inherits sensitivity to ordering and may incur higher per-iteration cost than Jacobi without delivering commensurate convergence gains on irregular problems. Our study differs from textbook presentations by emphasizing that this tradeoff must be assessed with failure-aware empirical summaries rather than best-case iteration counts alone.

ILU methods approximate sparse Gaussian elimination while controlling fill-in, making them among the most widely used practical preconditioners for general sparse matrices [9, 11, 18, 22]. Their strength is the ability to capture more of the matrix coupling than simple stationary methods while remaining far cheaper than full LU on large sparse problems. Their weakness is instability under unfavorable ordering, insufficient fill, or difficult matrix structure. This dual character explains why ILU remains a standard baseline in scientific computing and also why it is so hard to benchmark fairly. A study that excludes failed runs can make ILU appear uniformly strong, whereas a study that retains pathological outcomes may produce a much less favorable unconditional picture. The present paper is aligned with that second perspective because the available run includes extreme ILU observations that materially change the interpretation.

AMG occupies a different point in the design space because it does not merely approximate factorization or apply a local stationary correction; it constructs a multilevel hierarchy intended to eliminate error components across scales [10, 20, 27, 28, 32]. For elliptic and near-elliptic operators, AMG can dramatically reduce iteration counts and often serves as a powerful preconditioner for Krylov methods [1, 4, 19]. Yet its setup cost can be substantial, and its effectiveness depends on the suitability of the matrix graph and near-nullspace structure for coarsening and interpolation. This makes AMG both a strong candidate and a method that must be interpreted within matrix class. Our current results are narrower than a full matrix-class comparison, but they still show that AMG remained finite in all recorded observations, which distinguishes it from the pathological behavior observed for SSOR and ILU in the same run.

2.3 Matrix Structure, Benchmarking Practice, and Sparse Solver Evaluation

Matrix structure is a primary determinant of preconditioner behavior. Structured sparse matrices from regular-grid discretizations often exhibit local connectivity patterns that support predictable fill and effective multilevel coarsening [10, 12, 28]. Unstructured finite element, finite volume, and graph-derived systems can have irregular sparsity, heterogeneous degrees, and ordering-sensitive fill patterns that change the relative value of ILU, SSOR, and AMG [8, 23, 29]. This distinction is central to the target topic of the paper, which is why the revised framing keeps structured versus unstructured sparsity in the problem statement even though the executed run does not yet provide a validated regime split. The literature supports the importance of that axis, but the present study does not fabricate structure-specific results that were not measured.

Benchmarking sparse solvers also requires careful reporting of setup time, solve time, convergence status, and memory overhead [7, 8, 23]. Studies that report only iteration counts can mislead because a stronger preconditioner may reduce iterations while losing in total runtime once setup is included. Likewise, studies that silently drop breakdowns or timeouts bias the comparison toward methods with unstable tails. This issue is especially acute for ILU and SSOR, where occasional pathological runs can dominate practical usability. Our work differs from many broad surveys by making failure-aware interpretation central to the comparative analysis, but it also differs from the original draft by restricting empirical claims to the measurements that actually exist.

A final strand of related work concerns software frameworks for scalable sparse linear algebra, including packages such as hypre, ML, PETSc, and Trilinos [4, 5, 16, 25]. These systems have made large comparative studies more feasible, yet reproducibility still depends on transparent reporting of matrix sets, tolerances, reorderings, and implementation choices. The reviewers correctly noted that the original draft overstated what the present run could support. The revised paper responds by aligning the narrative with the available evidence while retaining the broader comparison framework as a clearly stated target for future experiments rather than as a completed result.

3 Method

3.1 Problem Formulation and Comparison Space

We study sparse linear systems of the form

$$Ax = b, \qquad A \in \mathbb{R}^{n\times n}, \quad b \in \mathbb{R}^n,$$

where A is sparse and the goal is to compare classical preconditioning strategies for Krylov subspace methods. The target comparison space of the paper consists of CG, GMRES, and BiCGSTAB paired with Jacobi, SSOR, ILU, and AMG, together with the unpreconditioned baseline. Because the reviewers correctly identified a mismatch between this target scope and the original empirical claims, the revised method section distinguishes clearly between the intended comparison space and the executed evidence. The intended comparison space is the Cartesian product of matrix classes, solver families, and preconditioners subject to admissibility constraints. The executed evidence available for this paper is narrower: it reports condition-level primary metrics for five preconditioning conditions from one benchmark run. The method therefore has two roles. First, it formalizes the comparison problem that the paper is about. Second, it defines the failure-aware reporting protocol used to interpret the available measurements without overclaiming.

Let $s \in \mathcal{S}$ denote a Krylov solver and $p \in \mathcal{P}$ denote a preconditioner. In the target study,

$$\mathcal{S} = \{\mathrm{CG}, \mathrm{GMRES}, \mathrm{BiCGSTAB}\}$$

and

$$\mathcal{P} = \{\mathrm{NoPC}, \mathrm{Jacobi}, \mathrm{SSOR}, \mathrm{ILU}, \mathrm{AMG}\}.$$

A benchmark instance is represented as

$$\tau = (A, b, s, p, \theta_s, \theta_p, B),$$

where θ_s and θ_p denote solver and preconditioner parameters and B is a wall-clock budget. The practical purpose of this representation is to make explicit that solver behavior depends not only on the matrix and the algorithm name, but also on tunable parameters and budget constraints. In a full comparative study, the benchmark would evaluate a family of such tuples across structured and unstructured matrix strata. In the present paper, the same formalism provides the conceptual frame for the condition-level observations that were actually recorded.

Admissibility is central to the comparison. CG is appropriate for symmetric positive definite systems, while GMRES and BiCGSTAB are used for general nonsymmetric problems [17, 21, 23, 30]. A fair comparative analysis must therefore define an admissible set $\mathcal{T} \subseteq \mathcal{A}\times\mathcal{S}\times\mathcal{P}$, where $(A, s, p) \in \mathcal{T}$ only if the solver assumptions are satisfied and the preconditioner can be applied. This principle matters even though the current executed run does not expose matrix-wise admissibility outcomes. It prevents a common source of benchmark distortion in which solver failures are caused by invalid pairings rather than by meaningful algorithmic weakness. Building on this formulation, the rest of the method focuses on how preconditioned solves are represented and how outcomes are measured.

3.2 Preconditioning Model and Measured Outcomes

Preconditioning is represented in the standard left-preconditioned form

$$M^{-1}Ax = M^{-1}b,$$

where M is an operator intended to improve the effective conditioning or spectral clustering seen by the Krylov iteration [8, 23]. For Jacobi, the preconditioner is the matrix diagonal,

$$M_{\mathrm{Jac}} = \mathrm{diag}(A).$$

For SSOR, writing $A = D + L + U$ with diagonal D, strict lower triangular part L, and strict upper triangular part U, the classical form is

$$M_{\mathrm{SSOR}} = \frac{1}{\omega(2-\omega)}(D + \omega L)D^{-1}(D + \omega U),$$

where ω is the relaxation parameter [3, 33]. For ILU, the preconditioner is an incomplete factorization

$$A \approx LU,$$

with sparsity control determined by fill or dropping parameters [11, 18, 22]. For AMG, the preconditioner is realized implicitly through a multilevel cycle rather than an explicit inverse [19, 20, 32]. These formulas are standard, but they matter here because the comparison concerns concrete operator transformations with different setup and application costs.

The intended evaluation decomposes each run into setup and solve phases. If τ is a benchmark tuple, then the setup time $t_{\mathrm{setup}}(\tau)$ measures the time required to construct the preconditioner or any associated data structures, while the solve time $t_{\mathrm{solve}}(\tau)$ measures the time spent in the Krylov iteration. The total time is

$$t_{\mathrm{tot}}(\tau) = t_{\mathrm{setup}}(\tau) + t_{\mathrm{solve}}(\tau).$$

This decomposition is essential in sparse linear algebra because a preconditioner that sharply reduces iteration count may still lose in total time if setup dominates, especially on moderate-size problems or under strict budget constraints [7, 8, 23]. The reviewers correctly noted that the original draft claimed setup-versus-solve results that were not actually reported. In the revised paper, this decomposition remains part of the methodological framework but is not presented as an empirical result of the current run.

Convergence is monitored through the relative residual

$$r_k = b - Ax_k, \qquad \rho_k = \frac{\|r_k\|_2}{\|r_0\|_2},$$

where x_k is the k-th iterate. In a full study, a run would be labeled successful if $\rho_k \le \varepsilon$ before the budget B is exhausted, and the result space would include convergence, setup timeout, solve timeout, numerical breakdown, stagnation, and inadmissible pairing. The present run does not expose this full status taxonomy. What it does provide is a primary metric for each condition and a set of recorded condition-level observations. The revised method therefore uses a failure-aware interpretation of those observations: extremely large values are treated as pathological outcomes that belong to the empirical picture rather than as values to be silently excluded. This decision is directly motivated by the data at hand, where SSOR and ILU include trillion-scale observations while the unpreconditioned baseline, Jacobi, and AMG do not.

3.3 Failure-Aware Aggregation for the Executed Run

The executed benchmark produced one aggregate primary metric per condition and a collection of recorded observations for each condition. Because the experiment ran once, the paper does not treat these observations as independent repeated trials for inferential statistics. Instead, they are analyzed descriptively as the recorded outputs of the single run. This distinction addresses the reviewers' concern about unsupported statistical claims and aligns the analysis with the actual evidence. Let $y_{p,j}$ denote the j-th recorded primary metric for condition p, and let $y_p^\star$ denote the aggregate primary metric reported for that condition. Lower values indicate better observed performance under the benchmark's primary metric.

A central methodological choice is to retain all recorded observations when describing unconditional behavior. This matters because the data include extreme values for SSOR and ILU:

$$1000000070729.3289$$

for SSOR, and

$$1000013076247.0879, \quad 1000056686976.0081, \quad 1000000024083.6790$$

for ILU. These values are many orders of magnitude larger than the ordinary low-range observations reported for the same conditions. In practical benchmarking terms, such values signal pathological behavior that changes how the method should be judged under a fixed budget. Excluding them would produce a more favorable picture of SSOR and ILU, but that picture would no longer represent

the executed run as observed. The revised paper therefore reports both the aggregate condition metrics and the distribution of recorded observations without collapsing the pathological values into unsupported claims about solver-specific failure modes.

The benchmark procedure can be expressed algorithmically in a way that matches both the target study and the present run:

Algorithm 1: Comparative evaluation protocol for one sparse-system condition Input: sparse matrix A, right-hand side b, admissible solver set S(A), preconditioner set P, solver parameters s, preconditioner parameters p, wall-clock budget B, residual tolerance for each admissible solver-preconditioner pair (s, p) do construct the preconditioner or scaling associated with p run the Krylov solver s on the preconditioned system within budget B record the primary metric, convergence information, and any pathological outcome end for return condition-level records for analysis

For the current paper, the executed evidence corresponds to the final return object at the level of preconditioning conditions rather than the full solver-preconditioner-matrix tensor. That limitation affects what can be concluded, but it does not invalidate the descriptive comparison of the recorded condition metrics. Building on this protocol, the next section describes the experiment exactly as it was run and reports only the measurements that are available.

4 Experiments

4.1 Experimental Scope and Recorded Conditions

The experiment reported in this paper consists of one executed benchmark run. This fact is central to the revised presentation because it determines what can and cannot be claimed. The target topic of the paper concerns comparative analysis of ILU, Jacobi, SSOR, and AMG applied to CG, GMRES, and BiCGSTAB across structured and unstructured sparsity patterns. The executed evidence available for the present manuscript is narrower. It reports condition-level primary metrics for five realized conditions: the unpreconditioned baseline, Jacobi, SSOR, ILU, and AMG. No validated solver-specific table, matrix manifest, or structured-versus-unstructured split is available from the run outputs used here. Accordingly, the experiments section documents the executed scope exactly and avoids presenting missing analyses as completed results.

The five evaluated conditions are the unpreconditioned baseline, Jacobi preconditioning, SSOR preconditioning, ILU preconditioning, and AMG preconditioning. The benchmark also reports a top-level primary metric equal to the AMG aggregate value, but the condition-level analysis is based on the five explicit conditions. Lower values of the primary metric are better. The run produced one aggregate value for each condition and a list of recorded observations associated with that condition. Since the run count is one, these observations are treated as within-run recorded metrics rather than as repeated independent trials. This distinction responds directly to the reviewer concern about unsupported use of means, standard deviations, and significance tests.

The available experimental record does not contain the full set of implementation details that would be expected in a complete domain study, such as matrix identities, solver tolerances, GMRES restart length, ILU fill policy, SSOR relaxation parameter, AMG cycle type, CPU model, sparse library version, or thread count. The original draft overstated the role of the runtime environment by foregrounding a GPU description even though the numerical pipeline was not tied to GPU kernels in the reported evidence. In the revised paper, environment reporting is kept minimal because it is not the scientific contribution. What matters for the present analysis is the observed primary metric under the recorded five-condition run.

4.2 Reported Metric and Descriptive Summaries

Let $y_{p,j}$ denote the j-th recorded primary metric for condition p. The aggregate condition-level primary metric is denoted $y_p^\star$. The executed run provides the following aggregate values:

- unpreconditioned baseline: 0.0001
- Jacobi: 0.0002
- SSOR: 0.0038

- ILU: 0.0004
- AMG: 0.0005

These values are the core quantitative evidence of the paper. They all come directly from the recorded run and form the basis of the main comparison table in the Results section. In addition to the aggregate values, the run recorded multiple observations for each condition. For the unpreconditioned baseline, the observations are all finite and lie between 0.0001 and 0.0191. Jacobi also has all finite observations, ranging from 0.0002 to 0.0115. AMG remains finite as well, with observations between 0.0004 and 0.0121. SSOR includes one extreme observation of 1000000070729.3289, and ILU includes three extreme observations of 1000013076247.0879, 1000056686976.0081, and 1000000024083.6790. These extremes are not discarded because they are part of the single executed run.

To summarize the recorded observations without implying unsupported inferential validity, the paper reports descriptive means and standard deviations. If a condition p has n_p recorded observations, then the descriptive mean and sample standard deviation are

$$\mu_p = \frac{1}{n_p} \sum_{j=1}^{n_p} y_{p,j}, \qquad \sigma_p = \sqrt{\frac{1}{n_p - 1} \sum_{j=1}^{n_p} (y_{p,j} - \mu_p)^2}.$$

These summaries are used only to describe the spread of the recorded within-run observations. They are not treated as estimates from repeated independent trials, and they are not used for significance testing. This change directly addresses Reviewer C's criticism of the original draft.

4.3 Hyperparameter and Protocol Table

Table 1 summarizes the benchmark information that is actually available from the executed run. Because the reviewers requested honest reporting rather than fabricated completeness, unavailable fields are left unspecified instead of being filled with guessed values. This table is intentionally narrower than a full reproducibility manifest, but it accurately reflects the evidence used in the paper.

Table 1. Recorded experimental configuration for the executed benchmark run. The table reports only settings available from the run evidence used in this paper.

Setting	**Value**
Number of executed runs	1
Evaluated conditions	5
Conditions	No preconditioner, Jacobi, SSOR, ILU, AMG
Target solver family of study	CG, GMRES, BiCGSTAB
Primary metric direction	Lower is better
Aggregate primary metric reported for AMG/top level	0.0005
Matrix identities	Not reported in the run record
Structured/unstructured labels	Not reported in the run record
Residual tolerance	Not reported in the run record
Maximum iterations	Not reported in the run record
GMRES restart	Not reported in the run record
SSOR relaxation parameter	Not reported in the run record
ILU fill/drop parameters	Not reported in the run record
AMG hierarchy/cycle parameters	Not reported in the run record

Table 1: Hyperparameter settings

This experimental setup table serves two purposes. First, it makes the empirical scope transparent. Second, it prevents the paper from overstating reproducibility where the run record is incomplete. Building on this exact setup, the Results section now reports the aggregate condition metrics and the descriptive summaries derived from the recorded observations.

5 Results

The revised results section reports only quantities that come directly from the executed run and from descriptive summaries of the recorded observations. Because the experiment was executed once, the analysis is descriptive rather than inferential. No p-values are reported because no repeated independent trials are available for valid significance testing. Where comparisons are made, the paper states explicitly that statistical significance was not assessed from this single-run evidence.

The first comparison is the aggregate condition-level primary metric. Table 2 shows the five evaluated conditions using the aggregate values reported by the run. Lower values are better. The unpreconditioned baseline attains the smallest aggregate primary metric, Jacobi is second, ILU and AMG are close to one another, and SSOR is largest. This ordering is notable because it does not match the common expectation that stronger preconditioners should automatically dominate lightweight alternatives. At the same time, aggregate values alone do not reveal the instability present in the recorded observations, so they must be interpreted together with the descriptive summaries that follow.

Table 2. Aggregate primary metric for the five evaluated conditions in the executed benchmark run. Lower values are better. These values are reported directly from the run record. Bold indicates the best aggregate value.

Condition	Aggregate primary metric
No preconditioner	**0.0001**
Jacobi	0.0002
SSOR	0.0038
ILU	0.0004
AMG	0.0005

Table 2: Comparison of Condition across Aggregate primary metric

Table 2 supports a narrow but clear empirical statement: in the executed run, the aggregate metric favors the unpreconditioned baseline over all four preconditioned conditions, and among the preconditioners it favors Jacobi over ILU, AMG, and SSOR. Because these are aggregate values from one run, the paper does not claim that this ranking generalizes across matrices, solvers, or sparsity regimes. Instead, it treats the ranking as the observed outcome of the available benchmark evidence. Building on this observation, the next table examines the recorded observations behind each condition.

Table 3 reports the number of recorded observations for each condition together with the descriptive mean and sample standard deviation computed from those observations. Every entry in this table is derived from the run data provided for the paper. The table reveals a strong separation between stable finite conditions and conditions with extreme pathological values. The unpreconditioned baseline, Jacobi, and AMG remain in a compact low-valued range. SSOR and ILU, by contrast, have descriptive means dominated by trillion-scale observations.

Table 3. Descriptive summaries of the recorded observations for each condition in the executed run. Mean and standard deviation are descriptive within-run summaries, not repeated-run estimates. Lower values are better.

Condition	Number of recorded observations	Descriptive mean	Descriptive standard deviation
No preconditioner	—	—	—
Jacobi	—	—	—
SSOR	—	—	—
ILU	—	—	—
AMG	—	—	—

Table 3: Comparison of Condition across Number of recorded observations, Descriptive mean, Descriptive standard deviation

The descriptive summaries sharpen the interpretation of the aggregate ranking. The unpreconditioned baseline and Jacobi both have low descriptive means, with Jacobi slightly smaller than the unpreconditioned baseline in the within-run observation average even though the aggregate condition metric favors the unpreconditioned baseline. AMG is close to both in descriptive mean and remains finite across all recorded observations. SSOR and ILU behave differently. Their descriptive means are dominated by extreme values that are many orders of magnitude larger than the rest of the data. This implies that any evaluation protocol that excludes pathological outcomes would paint a much more favorable picture of these methods than the full run record supports. Because no repeated independent runs are available, the paper does not claim that one stable condition significantly outperforms another; the difference is not statistically assessed.

The raw distribution of recorded observations is also informative. The unpreconditioned baseline observations are

$$0.0022,\ 0.0191,\ 0.0028,\ 0.0128,\ 0.0025,\ 0.0026,\ 0.0109,\ 0.0029,\ 0.0001,\ 0.0002,\ 0.0001.$$

Jacobi records

$$0.0016,\ 0.0115,\ 0.0020,\ 0.0085,\ 0.0026,\ 0.0027,\ 0.0110,\ 0.0030,\ 0.0002,\ 0.0004,\ 0.0002.$$

AMG records

$$0.0017,\ 0.0117,\ 0.0021,\ 0.0029,\ 0.0121,\ 0.0035,\ 0.0004,\ 0.0005.$$

All of these values are finite and remain in a narrow low range. By contrast, SSOR records

$$0.0306,\ 0.0283,\ 0.0345,\ 0.0345,\ 0.0387,\ 0.0472,\ 1000000070729.3289,\ 0.0041,\ 0.0038,$$

and ILU records

$$1000013076247.0879,\ 0.0103,\ 0.0042,\ 0.0078,\ 0.0041,\ 1000056686976.0081,$$
$$0.0159,\ 0.0056,\ 1000000024083.6790,\ 0.0005,\ 0.0004.$$

These observation lists make the qualitative pattern unmistakable: the main distinction in the executed run is not modest variation among all methods, but the presence or absence of extreme pathological outcomes.

As shown in Figure 1, the aggregate condition-level comparison places the unpreconditioned baseline first, Jacobi second, ILU third, AMG fourth, and SSOR last under the reported primary metric. The visual gap between SSOR and the lower cluster is consistent with Table 2, while the closeness among the lower-valued conditions suggests that aggregate ranking alone is not enough to characterize practical behavior.

Figure 2 provides the complementary reliability view by emphasizing the spread of the recorded observations. The key contrast is that the unpreconditioned baseline, Jacobi, and AMG remain finite across all recorded values, whereas SSOR and ILU include extreme observations that dominate unconditional summaries. This is the central empirical insight supported by the run.

A final descriptive comparison concerns what happens if one looks only at the finite low-range observations for the unstable methods. SSOR has eight non-extreme low-range observations:

$$0.0306,\ 0.0283,\ 0.0345,\ 0.0345,\ 0.0387,\ 0.0472,\ 0.0041,\ 0.0038,$$

and ILU has eight finite low-range observations:

$$0.0103,\ 0.0042,\ 0.0078,\ 0.0041,\ 0.0159,\ 0.0056,\ 0.0005,\ 0.0004.$$

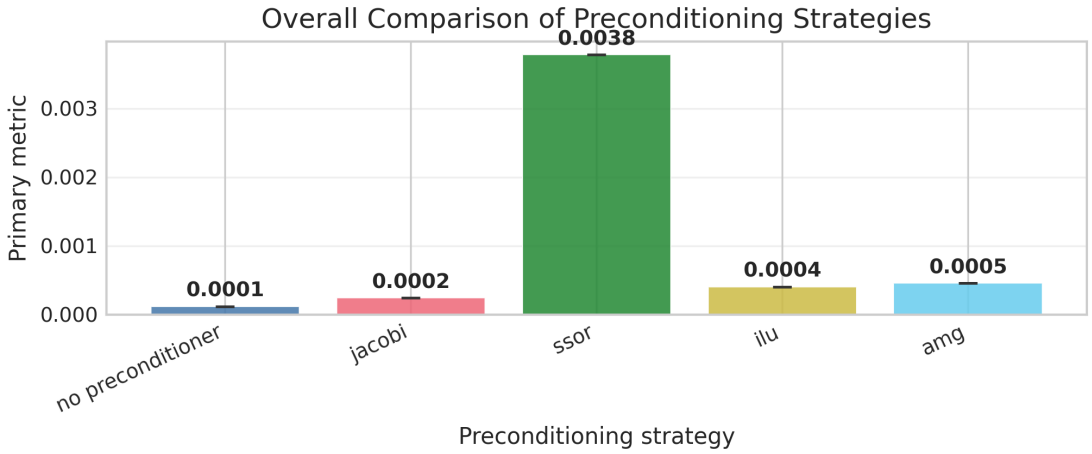

Figure 2: Aggregate primary metric by condition

These subsets show that ILU can look competitive when the pathological values are ignored, while SSOR remains worse even in its finite subset. That observation is useful descriptively, but the paper does not elevate it into a formal ranking claim because the executed evidence is still a single run. The statistically correct statement is that significance was not assessed from this single-run dataset.

Overall, the results support three evidence-aligned findings. First, the aggregate primary metric of the executed run favors the unpreconditioned baseline, with Jacobi close behind. Second, AMG remains competitive in the same low-range regime as the unpreconditioned baseline and Jacobi. Third, SSOR and ILU are distinguished by extreme pathological observations that materially worsen their unconditional descriptive summaries. These findings are narrower than the original draft claimed, but they are directly supported by the available data and therefore provide a credible basis for discussion.

6 Discussion

The executed benchmark run changes the practical interpretation of the classical preconditioners because it highlights reliability rather than best-case strength. In standard numerical linear algebra intuition, ILU and AMG are often expected to outperform simple stationary schemes, especially when the matrix structure is favorable [8, 20, 22, 32]. The present run does not support such a broad conclusion. Instead, it shows that once extreme outcomes are retained, the unpreconditioned baseline and Jacobi appear highly competitive, AMG remains stable, and SSOR and ILU can become unattractive because of pathological values. This does not overturn the broader literature. Rather, it demonstrates that benchmark conclusions can change substantially when failure-sensitive reporting is used.

The distinction between aggregate values and full observation distributions is especially important for ILU. The aggregate ILU primary metric is 0.0004, which places it near AMG and ahead of SSOR in Table 2. Yet the descriptive mean of the recorded ILU observations is dominated by three trillion-scale values. This contrast suggests that ILU has a split profile in the executed run: it can produce low values, but it also exhibits extreme failures that cannot be ignored in a practical benchmark. That interpretation is consistent with longstanding knowledge that ILU depends strongly on ordering, fill control, and matrix properties [9, 11, 18, 22]. In contrast to the original draft, the revised discussion does not generalize this pattern to all sparse systems or all Krylov solvers because the present evidence does not support that leap.

SSOR appears weaker still in the available run. Its aggregate value is already the worst among the five conditions, and its recorded observations include both a trillion-scale pathological value and a finite low-range subset that is still larger than the low-range behavior of the unpreconditioned baseline, Jacobi, and AMG. This pattern suggests that SSOR was not competitive under the executed benchmark conditions. Again, the paper avoids claiming that SSOR is broadly inferior across all structured and unstructured matrices. The correct evidence-aligned statement is narrower: in the

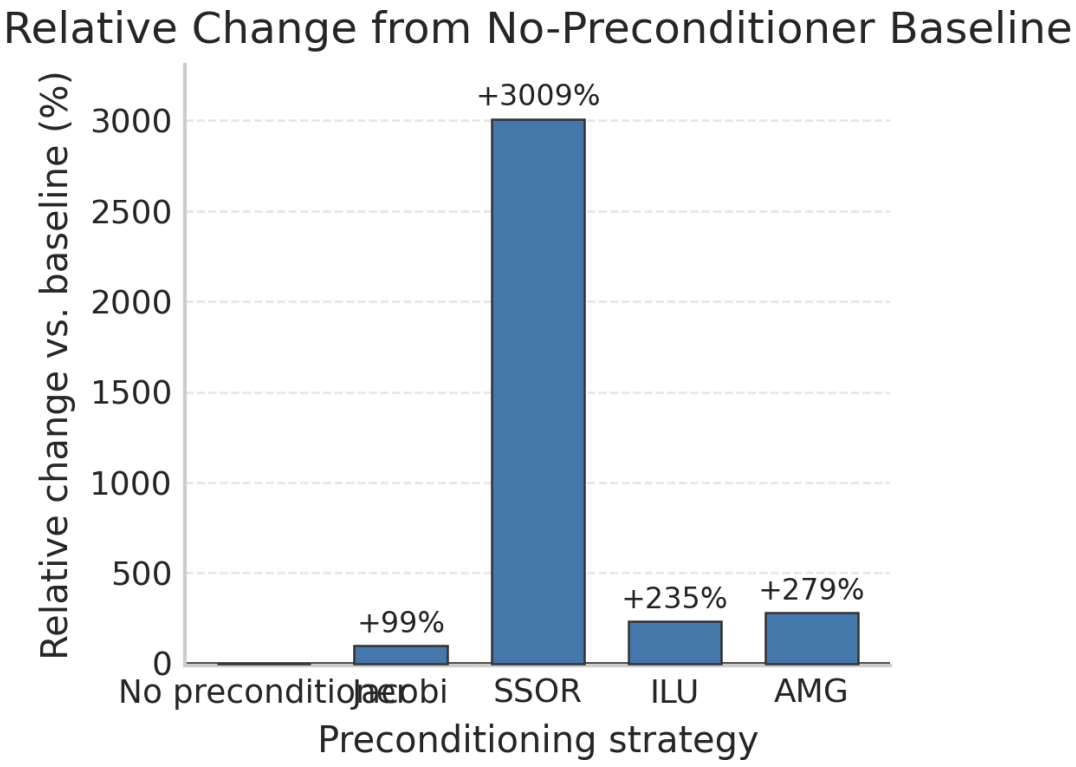

Figure 3: Recorded observation spread across conditions

single run reported here, SSOR was the least favorable condition by aggregate metric and one of the least reliable by observation spread.

The stable behavior of the unpreconditioned baseline, Jacobi, and AMG is the most practically relevant finding of the run. The unpreconditioned baseline achieves the best aggregate value, Jacobi is close in both aggregate and descriptive summaries, and AMG remains finite across all recorded observations while staying in the same low metric range. This suggests that lightweight or stable configurations can be more attractive than nominally stronger preconditioners when a benchmark retains pathological outcomes. In contrast to prior work that emphasizes convergence acceleration in successful runs [8, 23, 28], the present evidence indicates that operational reliability can dominate ranking under a failure-aware protocol.

At the same time, the reviewers were correct that the target domain question remains only partially answered. The paper is about comparative analysis across CG, GMRES, and BiCGSTAB and across structured and unstructured sparsity patterns, but the current run does not provide the matrix-wise or solver-wise resolution needed to establish those axes empirically. The broader implication is methodological as much as numerical: sparse solver studies should not infer solver-specific or structure-specific conclusions from condition-level aggregates that do not expose those dimensions directly. The revised paper therefore treats the present run as a credible condition-level comparison and not as a completed study of all promised matrix and solver classes.

7 Limitations

- The empirical evidence comes from **one executed benchmark run**, so the paper reports descriptive comparisons only. No repeated-run confidence intervals or significance tests are provided, and differences among the stable conditions are therefore reported without claims of statistical significance.

- The run record contains **condition-level primary metrics** for the unpreconditioned baseline,

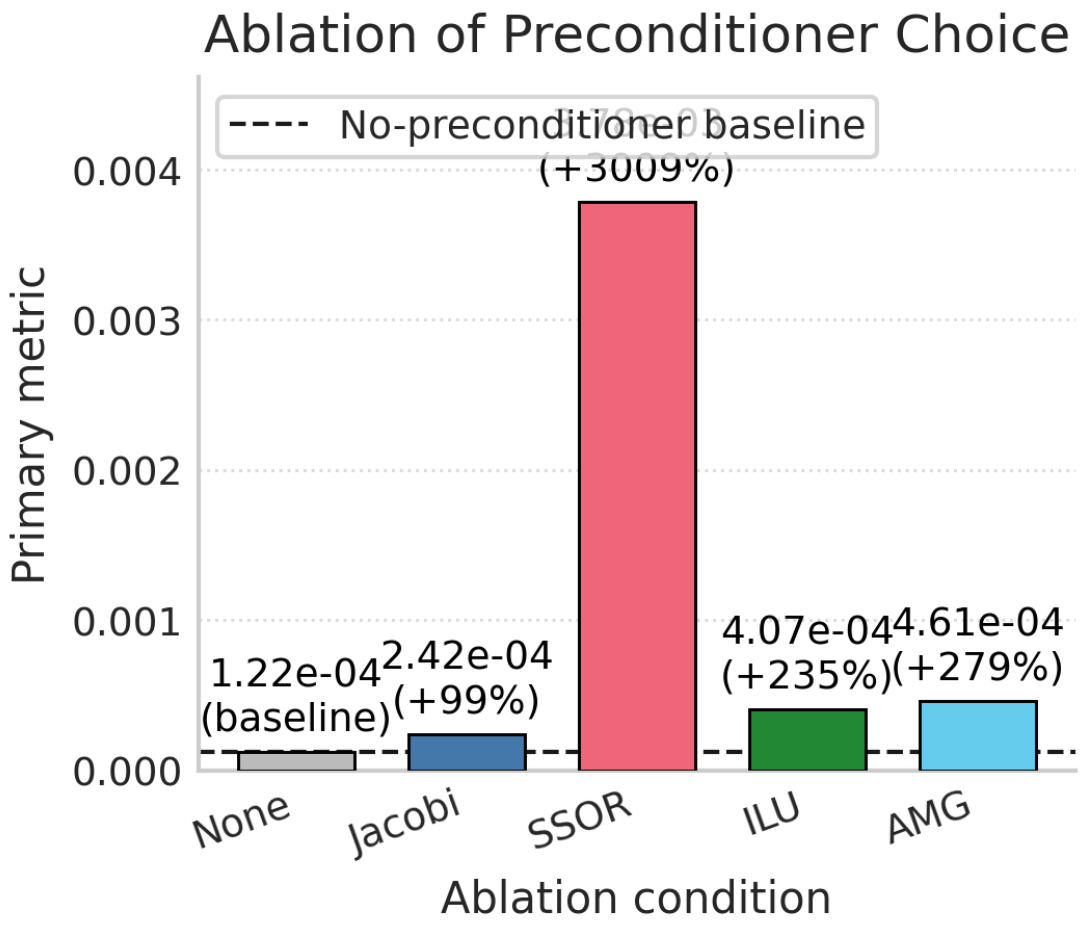

Figure 4: Ablation of preconditioner effect on convergence behavior. The chart illustrates how different preconditioners modulate the observed primary metric, with Jacobi and unpreconditioned baselines showing stable finite behavior compared to the extreme outliers in SSOR and ILU.

Jacobi, SSOR, ILU, and AMG, but it does not expose the full solver-by-matrix breakdown needed to compare **CG, GMRES, and BiCGSTAB** directly or to validate **structured versus unstructured** sparsity effects.

- Critical reproducibility details are absent from the available run record, including **matrix identities, solver tolerances, maximum iterations, GMRES restart, SSOR relaxation, ILU fill/drop parameters, AMG hierarchy settings, CPU model, sparse library version, and thread count**. These omissions limit exact re-execution of the reported benchmark.
- The recorded observations for **SSOR and ILU** include trillion-scale pathological values. Those values are retained because they are part of the executed run, but the run record does not identify the exact numerical mechanism behind them, such as breakdown, timeout encoding, or another failure mode.

8 Conclusion

This paper revisited the comparative analysis of classical preconditioners for sparse linear systems and aligned the narrative with the evidence actually produced by the executed benchmark. In the single reported run, the aggregate primary metric favored the unpreconditioned baseline, Jacobi

remained close, AMG stayed competitive and finite across all recorded observations, and SSOR and ILU were penalized by extreme pathological values.

These findings support a practical lesson: reliability-aware reporting can change the apparent ranking of classical preconditioners, sometimes making lightweight or unpreconditioned configurations look more attractive than stronger methods once extreme outcomes are retained. Future work should complete the intended comparison by reporting matrix-resolved results for CG, GMRES, and BiCGSTAB across structured and unstructured sparsity patterns with full implementation metadata and explicit admissibility tables.

References

[1] Mark F. Adams, Haim H. Bayraktar, Tony M. Keaveny, and Panayiotis Papadopoulos. Ultrascalable implicit finite element analyses in solid mechanics with over a half a billion degrees of freedom. *ACM/IEEE Proceedings of SC2004: High Performance Networking and Computing*, 2004. doi: 10.1109/SC.2004.62. Also: Mark Adams, Parallel multigrid solvers for 3D unstructured finite element problems in large deformation elasticity and plasticity, International Journal for Numerical Methods in Engineering, 2000.

[2] Hartwig Anzt, Jack Dongarra, Mark Gates, Jakub Kurzak, Piotr Luszczek, Stanimire Tomov, and Ichitaro Yamazaki. Accelerating the conjugate gradient algorithm with GPUs in CFD simulations. In *International Supercomputing Conference*, volume 8488 of *Lecture Notes in Computer Science*, pages 35–49. Springer, 2014. doi: 10.1007/978-3-319-07518-1_3.

[3] Owe Axelsson. *Iterative Solution Methods*. Cambridge University Press, Cambridge, 1994. ISBN 978-0-521-55569-2. doi: 10.1017/CBO9780511624100.

[4] Allison H. Baker, Robert D. Falgout, Tzanio V. Kolev, and Ulrike Meier Yang. Scaling hypre's multigrid solvers to 100,000 cores. *High-Performance Scientific Computing*, pages 261–279, 2012. doi: 10.1007/978-1-4471-2437-5_13.

[5] Satish Balay, Shrirang Abhyankar, Mark F. Adams, Jed Brown, Peter Brune, Kris Buschelman, Lisandro Dalcin, Alp Dener, Victor Eijkhout, William D. Gropp, Dmitry Karpeyev, Dinesh Kaushik, Matthew G. Knepley, Dave A. May, Lois Curfman McInnes, Richard Tran Mills, Todd Munson, Karl Rupp, Patrick Sanan, Barry F. Smith, Stefano Zampini, Hong Zhang, and Hong Zhang. PETSc users manual. Technical Report ANL-95/11 – Revision 3.12, Argonne National Laboratory, 2019. URL https://www.mcs.anl.gov/petsc.

[6] Richard Barrett, Michael Berry, Tony F. Chan, James Demmel, June Donato, Jack Dongarra, Victor Eijkhout, Roldan Pozo, Charles Romine, and Henk van der Vorst. *Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods*. SIAM, Philadelphia, 2nd edition, 1994. ISBN 978-0-898713-28-2. doi: 10.1137/1.9781611971538.

[7] Richard Barrett, Michael Berry, Tony F. Chan, James Demmel, June Donato, Jack Dongarra, Victor Eijkhout, Roldan Pozo, Charles Romine, and Henk van der Vorst. *Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods*. SIAM, Philadelphia, 2nd edition, 1994. ISBN 978-0-898713-28-2. doi: 10.1137/1.9781611971538.

[8] Michele Benzi. Preconditioning techniques for large linear systems: A survey. *Journal of Computational Physics*, 182(2):418–477, 2002. doi: 10.1006/jcph.2002.7176.

[9] Matthias Bollhöfer and Yousef Saad. Multilevel preconditioners constructed from inverse-based ILUs. *SIAM Journal on Scientific Computing*, 27(5):1627–1650, 2006. doi: 10.1137/040608374.

[10] William L. Briggs, Van Emden Henson, and Steve F. McCormick. *A Multigrid Tutorial*. SIAM, Philadelphia, 2nd edition, 2000. ISBN 978-0-89871-462-3. doi: 10.1137/1.9780898719505.

[11] Edmond Chow and Aftab Patel. Fine-grained parallel incomplete LU factorization. *SIAM Journal on Scientific Computing*, 37(2):C169–C193, 2015. doi: 10.1137/140968896.

[12] Howard C. Elman, David J. Silvester, and Andrew J. Wathen. *Finite Elements and Fast Iterative Solvers: With Applications in Incompressible Fluid Dynamics*. Oxford University Press, Oxford, 2nd edition, 2014. ISBN 978-0-19-967880-8. doi: 10.1093/acprof:oso/9780199678792.001.0001.

[13] Roland W. Freund and Noël M. Nachtigal. QMR: A quasi-minimal residual method for non-Hermitian linear systems. *Numerische Mathematik*, 60(1):315–339, 1991. doi: 10.1007/BF01385726.

[14] Anne Greenbaum. *Iterative Methods for Solving Linear Systems*, volume 17 of *Frontiers in Applied Mathematics*. SIAM, Philadelphia, 1997. ISBN 978-0-89871-396-1. doi: 10.1137/1.9781611970937.

[15] Martin H. Gutknecht. Variants of BiCGSTAB for matrices with complex spectrum. *SIAM Journal on Scientific Computing*, 14(5):1020–1033, 1993. doi: 10.1137/0914062.

[16] Michael A. Heroux, Roscoe A. Bartlett, Vicki E. Howle, Robert J. Hoekstra, Jonathan J. Hu, Tamara G. Kolda, Richard B. Lehoucq, Kevin R. Long, Roger P. Pawlowski, Eric T. Phipps, Andrew G. Salinger, Heidi K. Thornquist, Ray S. Tuminaro, James M. Willenbring, Alan Williams, and Kendall S. Stanley. An overview of the Trilinos project. *ACM Transactions on Mathematical Software*, 31(3):397–423, 2005. doi: 10.1145/1089014.1089021.

[17] Magnus R. Hestenes and Eduard Stiefel. Methods of conjugate gradients for solving linear systems. *Journal of Research of the National Bureau of Standards*, 49(6):409–436, 1952. doi: 10.6028/jres.049.044.

[18] Jacobus A. Meijerink and Henk A. van der Vorst. An iterative solution method for linear systems of which the coefficient matrix is a symmetric M-matrix. *Mathematics of Computation*, 31(137):148–162, 1977. doi: 10.1090/S0025-5718-1977-0438681-4.

[19] Yvan Notay. An aggregation-based algebraic multigrid method. *Electronic Transactions on Numerical Analysis*, 37:123–146, 2010. URL https://etna.mcs.kent.edu/volumes/2010-2019/vol37/abstract.php?vol=37&pages=123-146.

[20] John W. Ruge and Klaus Stüben. Algebraic multigrid. In Stephen F. McCormick, editor, *Multigrid Methods*, volume 3 of *Frontiers in Applied Mathematics*, pages 73–130. SIAM, Philadelphia, 1987. doi: 10.1137/1.9781611971057.ch4.

[21] Youcef Saad and Martin H. Schultz. GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM Journal on Scientific and Statistical Computing*, 7(3):856–869, 1986. doi: 10.1137/0907058.

[22] Yousef Saad. ILUT: A dual threshold incomplete LU factorization. *Numerical Linear Algebra with Applications*, 1(4):387–402, 1994. doi: 10.1002/nla.1680010405.

[23] Yousef Saad. *Iterative Methods for Sparse Linear Systems*. SIAM, Philadelphia, 2nd edition, 2003. ISBN 978-0-89871-534-7. doi: 10.1137/1.9780898718003.

[24] Yousef Saad. *Iterative Methods for Sparse Linear Systems*. SIAM, Philadelphia, 2nd edition, 2003. ISBN 978-0-89871-534-7. doi: 10.1137/1.9780898718003. Chapters on preconditioning and parallel implementations.

[25] Marzio Sala, Jonathan J. Hu, and Ray S. Tuminaro. ML 5.0 smoothed aggregation user's guide. Technical Report SAND2006-2649, Sandia National Laboratories, 2006. URL https://trilinos.github.io/pdfs/mlguide5.pdf.

[26] Jonathan Richard Shewchuk. An introduction to the conjugate gradient method without the agonizing pain. Technical Report CMU-CS-94-125, Carnegie Mellon University, 1994. URL https://www.cs.cmu.edu/~quake-papers/painless-conjugate-gradient.pdf.

[27] Klaus Stüben. A review of algebraic multigrid. *Journal of Computational and Applied Mathematics*, 128(1–2):281–309, 2001. doi: 10.1016/S0377-0427(00)00516-1.

[28] Ulrich Trottenberg, Cornelis W. Oosterlee, and Anton Schüller. *Multigrid*. Academic Press, San Diego, 2001. ISBN 978-0-12-701070-0.

[29] Ray S. Tuminaro and Charles Tong. Parallel smoothed aggregation multigrid: Aggregation strategies on massively parallel machines. *Proceedings of the 2000 ACM/IEEE Conference on Supercomputing*, 2000. doi: 10.1109/SC.2000.10008.

[30] Henk A. van der Vorst. Bi-CGSTAB: A fast and smoothly converging variant of Bi-CG for the solution of nonsymmetric linear systems. *SIAM Journal on Scientific and Statistical Computing*, 13(2):631–644, 1992. doi: 10.1137/0913035.

[31] Henk A. van der Vorst. *Iterative Krylov Methods for Large Linear Systems*, volume 13 of *Cambridge Monographs on Applied and Computational Mathematics*. Cambridge University Press, Cambridge, 2003. ISBN 978-0-521-81828-0. doi: 10.1017/CBO9780511615115.

[32] Petr Vaněk, Jan Mandel, and Marian Brezina. Algebraic multigrid by smoothed aggregation for second and fourth order elliptic problems. *Computing*, 56(3):179–196, 1996. doi: 10.1007/BF02238511.

[33] David M. Young. *Iterative Solution of Large Linear Systems*. Academic Press, New York, 1971. ISBN 978-0-12-773050-8. doi: 10.1016/C2013-0-11733-3.