IMA Journal of Numerical Analysis (XXXX) **XX**, 1–44
https://doi.org/DOI HERE

Mixed precision strategies for preconditioned GMRES: a comprehensive analysis

ALFREDO BUTTARI
IRIT, CNRS, F-31071 Toulouse, France

XIN LIU
Academy of Mathematics and Systems Science, CAS, 100190 Beijing, China

THEO MARY
Sorbonne Université, LIP6, CNRS, F-75005 Paris, France

AND

BASTIEN VIEUBLÉ*
Academy of Mathematics and Systems Science, CAS, 100190 Beijing, China
*Corresponding author: bastien.vieuble@amss.ac.cn

[Received on Date Month Year; revised on Date Month Year; accepted on Date Month Year]

The Generalized Minimal Residual method (GMRES) for the solution of general square linear systems $Ax = b$ is often combined with a preconditioner to improve the convergence speed of the method. Successful mixed precision implementations for the application of the preconditioner inside GMRES have been previously proposed: certain strategies prescribe to apply the preconditioner in low precision to reduce overall time and memory consumption, and other strategies propose to apply the matrix A and the preconditioner in higher precision to improve robustness and accuracy. These existing studies tend to focus on one kind of preconditioner combined with one kind of preconditioning technique (left-, right-, and flexible-preconditioning). In this article, we wish to unify most of the state-of-the-art mixed precision implementations for preconditioned GMRES under the same comprehensive theory, give a clear and exhaustive list of the possible strategies to set the precisions, and explain how these strategies compare numerically. To achieve this, we derive rounding error analyses with generic preconditioners for the left-, right-, and flexible-preconditioned GMRES processes in mixed precision and obtain descriptive bounds for the attainable forward errors of the computed solutions. Specifically, we substantially improve the sharpness of the right- and flexible-preconditioned GMRES forward error bounds compared with the existing literature. From the study of these bounds, we discover new meaningful mixed precision implementations that were not previously known; these new strategies achieve new tradeoffs between the employment of computationally effective low precision and accuracy. Moreover, we also uncover critical differences in robustness and accuracy between left-, right-, and flexible-preconditioning for a same given set of precisions: the choice among the three preconditioning techniques therefore has higher stakes in mixed precision. We substantiate our theoretical findings with a comprehensive experimental study on random dense and real-life sparse matrices from the SuiteSparse collection with various preconditioners: low precision LU factorization, incomplete LU, sparse approximate inverse, and polynomial preconditioners.

Keywords: iterative solvers; mixed precision; preconditioner; GMRES; rounding errors; linear system of equations

1. Introduction

A popular choice of Krylov-based iterative solvers for the efficient solution of square nonsingular linear systems

$$Ax = b, \qquad A \in \mathbb{R}^{n\times n}, \qquad 0 \neq b \in \mathbb{R}^n, \tag{1.1}$$

is the Generalized Minimal Residual method [32] (GMRES). As for most Krylov-based iterative solvers, a critical element to help reduce the number of iterations of GMRES is preconditioning [36]. It consists in transforming the original linear system (1.1) into one which shares the same solution but which is easier to solve:

$$\text{(left)} \quad M_L^{-1}Ax = M_L^{-1}b \qquad \text{or} \qquad AM_R^{-1}x^R = b \quad \text{with} \quad x^R = M_R x \quad \text{(right)}, \tag{1.2}$$

where $M_L \in \mathbb{R}^{n\times n}$ and $M_R \in \mathbb{R}^{n\times n}$ are nonsingular. Depending on which preconditioned system is solved in (1.2), we refer to the method as left- or right-preconditioned GMRES.

Over the past years, GMRES has substantially benefited from the increasing availability of low precisions in supercomputers, which are often accessible in accelerators like GPUs and have opened up new opportunities for resource savings. Unfortunately, employing low precision tends to introduce inexactness in the computation. To counterbalance this effect, mixed precision algorithms compute strategic parts of the computation with high precision to preserve or recover accuracy. Mixed precision has been widely employed within GMRES and has proven to be effective. Some of the earliest studies on the topic date back to the 90s [34]. Since then, the literature around mixed precision Krylov solvers, and specifically GMRES, has flourished. A popular and successful mixed precision strategy is to employ an inner-outer layout, where the inner GMRES solver is operated in low precision and the outer iterative solver in high precision to improve accuracy cheaply. For instance, combining GMRES with an outer iterative refinement process in mixed precision has been extensively studied; see [6], [26], [28], [15], [3], [4], or [12]. Another example of such a mixed precision strategy is proposed in [10] where both inner and outer solvers are GMRES. Other mixed precision strategies prescribe to compute some GMRES operations and kernels in different precisions: mixed precision in the orthogonalization process [37], in the SpMV [20], for storing the basis [2][1], or for storing the preconditioner [5]. Lastly, some strategies change the precisions with which GMRES is performed as the iterations go. The work [33], [19], or [21] demonstrated that the precisions for computing the matrix–vector and scalar products could be reduced as we lose orthogonality on the computed basis. Conversely, [29] prescribes to increase progressively the GMRES precision between restarts if the accuracy of the computed solution stagnates or diverges. Note that these different mixed precision strategies are not necessarily exclusive and can be combined to some extent.

In this article, we reduce the scope of mixed precision implementations to the simple yet generic layout outlined in Algorithm 1. This layout considers preconditioned GMRES using Modified Gram-Schmidt (MGS) orthonormalization, and assumes that the application of the preconditioner to vectors is performed in precision u_m, the matrix–vector product with A is performed in precision u_a, and the rest of the GMRES operations are performed in precision u_g. We represent the left- and right-preconditioned variants of this algorithm, respectively, on the left-side and right-side of Algorithm 1. In addition to left- and right-preconditioning, we will also consider another preconditioning technique called flexible-preconditioned GMRES [31], which is a reformulation of the right-preconditioned GMRES where the vectors z_j in Algorithm 1 are stored persistently in memory in an additional basis $Z_k = [z_1, \ldots, z_k]$. In this case, lines 17 and 18 of Algorithm 1 (right) are computed as $x_k = x_0 + Z_k y_k$ instead. At the additional cost of storing another set of k vectors in memory, flexible-preconditioning allows the preconditioner

Algorithm 1 Preconditioned MGS-GMRES in mixed precision.

Input: an $n \times n$ matrix A and a preconditioner M, a right-hand side b, and a number of iteration k.
Output: a computed solution to $Ax = b$.

1:	Initialize x_0 (e.g., $x_0 = M^{-1}b$)	
2:	$\boldsymbol{r_0 = Ax_0 - b}$	$\boldsymbol{u_a}$
3:	$\boldsymbol{s_0 = M^{-1}r_0}$	$\boldsymbol{u_m}$
4:	$\beta = \|s_0\|_2$, $v_1 = s_0/\beta$, $V_1 = [v_1]$	u_g
5:	**for** $j = 1 : k$ **do**	
6:	$\quad \boldsymbol{w'_j = Av_j}$	$\boldsymbol{u_a}$
7:	$\quad \boldsymbol{w_j = M^{-1}w'_j}$	$\boldsymbol{u_m}$
8:	$\quad$ **for** $l = 1 : j$ **do**	
9:	$\quad\quad h_{l,j} = v_l^T w_j$	u_g
10:	$\quad\quad w_j = w_j - h_{l,j} v_l$	u_g
11:	$\quad$ **end for**	
12:	$\quad h_{j+1,j} = \|w_j\|_2$	u_g
13:	$\quad v_{j+1} = w_j / h_{j+1,j}$	u_g
14:	$\quad V_{j+1} = [V_j,\ v_{j+1}]$	
15:	**end for**	
16:	$y_k = \arg\min_y \|\beta e_1 - \bar{H}_k y\|_2$	u_g
17:		
18:	$x_k = x_0 + V_k y_k$	u_g

1:	Initialize x_0 (e.g., $x_0 = M^{-1}b$)	
2:	$\boldsymbol{r_0 = Ax_0 - b}$	$\boldsymbol{u_a}$
3:		
4:	$\beta = \|r_0\|_2, \quad v_1 = r_0/\beta, \quad$ j=0	u_g
5:	**for** $j = 1 : k$ **do**	
6:	$\quad \boldsymbol{z_j = M^{-1}v_j}$	$\boldsymbol{u_m}$
7:	$\quad \boldsymbol{w_j = Az_j}$	$\boldsymbol{u_a}$
8:	$\quad$ **for** $l = 1 : j$ **do**	
9:	$\quad\quad h_{l,j} = v_l^T w_j$	u_g
10:	$\quad\quad w_j = w_j - h_{l,j} v_l$	u_g
11:	$\quad$ **end for**	
12:	$\quad h_{j+1,j} = \|w_j\|_2$	u_g
13:	$\quad v_{j+1} = w_j / h_{j+1,j}$	u_g
14:	$\quad V_{j+1} = [V_j,\ v_{j+1}]$	
15:	**end for**	
16:	$y_k = \arg\min_y \|\beta e_1 - \bar{H}_k y\|_2$	u_g
17:	$y'_k = V_k y_k$	u_g
18:	$\boldsymbol{x_k = x_0 + M^{-1}y'_k}$	$\boldsymbol{u_m}$

M_R to vary from one iteration to another. In this study, however, the preconditioner M_R is fixed for all iterations, and flexible-preconditioned GMRES is equivalent in exact arithmetic to right-preconditioned GMRES. Yet, even if both are equivalent in exact arithmetic, the former can offer extra stability in inexact arithmetic, as remarked in [8].

Earlier work already assessed some combinations of the precisions u_m, u_a, and u_g in Algorithm 1; we will address them exhaustively in section 4.2 when presenting the associated strategies for choosing the precisions. However, many of these earlier studies are dedicated to one specific kind of preconditioner and one specific kind of preconditioning technique (left-, right-, or flexible-preconditioning). In particular, they often do not make it clear whether the proposed mixed precision strategies would be viable for other kinds of preconditioners or preconditioning techniques. They also do not relate or compare their given mixed precision strategy for choosing u_m, u_a, and u_g to other existing ones, and, as we will show, they only account for a limited subset of the possible combinations of the precisions u_m, u_a, and u_g with the three preconditioning techniques.

In this context, the core purpose of this article is to answer the question: what are (all) the numerically meaningful ways to set u_m, u_a, and u_g for left-, right-, and flexible-preconditioned GMRES and how do they compare? Here, "numerically meaningful" means that a given combination of these precisions should present a tradeoff between accuracy, number of iterations, and the employment of computationally effective low precision. Ultimately, we aim for this article to stand as a practical guide that lists all meaningful options for setting the precisions u_m, u_a, and u_g in Algorithm 1. To achieve this, we introduce our set of notations and mathematical tools which will be used throughout the article

in section 2. In section 3, we proceed to the rounding error analysis of left-, right-, and flexible-preconditioned GMRES to derive bounds for the attainable forward errors. To this end, we rework a key result of [11] which provides modular and generic error bounds for GMRES algorithms. Using this rework, we derive descriptive bounds using generic preconditioners. Specifically, our analysis leads to substantial improvements for the right- and flexible-preconditioned GMRES forward error bounds compared with the existing literature. In section 4, we use these bounds combined with numerical experiments on random dense matrices to identify meaningful mixed precision strategies; each strategy prescribes how to set u_m, u_a, and u_g to achieve a specific tradeoff between accuracy and employment of low precision. We recover existing combinations of the precisions u_m, u_a, and u_g already introduced in the literature, we uncover new ones, and we study and compare them for left-, right-, and flexible-preconditioned GMRES. In particular, we unveil major differences in robustness and accuracy for each preconditioning technique given a same combination of precisions. We conclude that left-, right-, and flexible-preconditioning present different strengths and weaknesses and that, in some cases, using one over the others is preferable or even critical. Finally, in section 5, we validate our findings with numerical experiments on real-life problems from the SuiteSparse collection [18] using a variety of commonly used preconditioners: incomplete LU factorization, low precision LU factorization, sparse approximate inverse, and polynomial preconditioners. We provide our concluding remarks in section 6.

We let the reader know that this article builds upon certain unpublished ideas presented in the PhD thesis [35, chap. 7] of the fourth author of this article. Some of these ideas have also been cited and extended in [13] which focuses on applying the preconditioners in mixed precision for split-preconditioned GMRES. We do not address the split-preconditioning case, and we recommend a reader interested in the topic to read [13], which naturally complements this article's content.

2. Notations

For convenience, the notations u_m, u_a, and u_g can refer to both the floating-point arithmetic or its unit roundoff, depending on the context. Throughout the rounding error analysis of section 3, these three precision parameters are unspecified and can represent any floating-point arithmetic as long as the associated unit roundoff is substantially lower than 1. Later in the experimental sections 4 and 5, we will assign u_m, u_a, and u_g to specific arithmetics. We list in Table 1 the ones we use; they are a sample of some of the widely accessible floating-point arithmetics in supercomputers.

TABLE 1 *Parameters for floating-point arithmetics: symbol used in this paper, number of bits for the significand (including the implicit leading bit), number of bits for the exponent, unit roundoff, and range.*

Arithmetic	Symbol	Significand	Exponent	Unit roundoff	Range
bfloat16	B	8	8	3.9×10^{-3}	$10^{\pm 38}$
fp32	S	24	8	6.0×10^{-8}	$10^{\pm 38}$
fp64	D	53	11	1.1×10^{-16}	$10^{\pm 308}$
fp128	Q	113	15	9.6×10^{-35}	$10^{\pm 4932}$

We use the notations M_L to refer to the left-preconditioner and M_R to refer to the right-preconditioner. When the context does not need to differentiate between left- and right-preconditioner,

we simply write M. Identically, we use the notation $\widetilde{A}$ to refer to the preconditioned matrix $M_L^{-1}A$ or AM_R^{-1}.

We use the standard model of floating-point arithmetic [23, sect. 2.2] and we use the notation $\mathrm{fl}(\cdot)$ to denote the computed value of a given expression.

Our analysis is a traditional worst case analysis and the error bounds obtained depend on some constants related to the problem dimension n and the number of GMRES iterations k. We gather these constants into generic functions $c(n,k)$. We guarantee that these functions $c(n,k)$ are polynomials in n and k of low degree, but since they are known to be pessimistic [24], we do not always keep track of their precise values.

We use the notations $\lesssim$ and $\approx$ when dropping negligible second order terms in the error bounds, and the notation $\Theta_1 \gg \Theta_2$ to indicate that Θ_1 is much greater than Θ_2. In particular, we consider that if $\Theta_1 \gg \Theta_2$, then we can safely assess that $\Theta_1 \gg c(n,k)\Theta_2$. We also use the notation $\equiv$, which means that we can take the quantity on the left, which is in our control and is not fixed, to be equal to the quantity on the right.

We write $\sigma_{\min}(B)$ and $\sigma_{\max}(B)$ for the smallest and largest singular value of a rectangular matrix $B \in \mathbb{R}^{n\times m}$, respectively. Our error analysis uses both the 2-norm and the Frobenius norm, denoted by $\|\cdot\|_2$ and $\|\cdot\|_F$. The 2-norm of a matrix $B \in \mathbb{R}^{n\times m}$ refers to the induced norm. We define the normwise condition numbers of a square nonsingular matrix $B \in \mathbb{R}^{n\times n}$ by $\kappa_F(B) = \|B^{-1}\|_F\|B\|_F$, $\kappa_2(B) = \sigma_{\max}(B)/\sigma_{\min}(B)$, and $\kappa_{F,2} = \|B\|_F/\sigma_{\min}(B)$. Because of the equivalence of the 2-norm and Frobenius norm, the context often does not require differentiating these quantities and we simply write $\kappa(B)$.

The forward error of a computed solution $\widehat{x}$ by Algorithm 1 of the linear system (1.1) is defined as

$$\frac{\|\widehat{x}-x\|_2}{\|x\|_2}. \tag{2.1}$$

3. Bounds on the attainable forward errors

The choice of preconditioner and precisions u_m, u_a, and u_g affects both the convergence rate and the attainable errors of GMRES. Regarding the convergence rate, a practical rule of thumb is to consider that the more the preconditioner reduces the condition number of the original matrix A and clusters the eigenvalues, the better the convergence. Hence, the ideal scenario is to obtain $\kappa_2(\widetilde{A}) \approx 1$ where M^{-1} is computed and/or applied cheaply. Unfortunately, because "any nonincreasing convergence curve is possible for GMRES" [22], there is actually very little we can theoretically ensure about how much a given preconditioner will improve the convergence. For this reason, our theoretical analysis will mostly focus on identifying how the choice of preconditioner and precisions affects the attainable forward error, and subsequently rely on these results to identify the meaningful strategies to set the precisions. Nevertheless, we will assess both the convergence rate and the attainable errors in the experimental sections.

Note that we focus our analysis solely on the forward error (2.1) of the linear system (1.1). Backward error [23, sec. 7.1] bounds can also be obtained for preconditioned GMRES, but they would require dedicated developments which we cannot add concisely to this article.

3.1. *Modular error bound*

To derive bounds on the attainable forward errors for left-, right-, and flexible-preconditioned GMRES, we wish to use the modular framework for the backward error analysis of GMRES developed in [11].

Particularly, we need to develop and call an improved version of [11, Thm. 3.1], which provides sharper bounds on the attainable forward errors of the preconditioned GMRES implementations we consider. Our rework of [11, Thm. 3.1] will be presented in the following Theorem 3.1 and, contrary to [11, Thm. 3.1], which can handle many orthogonalization methods, will be specialized to GMRES employing MGS orthogonalization as in Algorithm 1. We will then use this result to study individually left-, right-, and flexible-preconditioned GMRES in sections 3.2, 3.3, and 3.4, respectively.

We follow a similar approach to [11] by remarking that Algorithm 1 can be compacted and rewritten under the form of Algorithm 2. In Algorithm 2, $\widetilde{Z}_k$ is an input and refers to the basis that spans the search space; that is, the space from which we extract the approximation x_k to the exact solution x. In exact arithmetic, we simply have $\widetilde{Z}_k \equiv Z_k = M_R^{-1}V_k$, where $V_k = [v_1, \ldots, v_k]$ is the orthogonal Krylov basis computed exactly. More specifically, V_k is constructed iteratively from the QR factorization $[\widetilde{b}, C_k] = V_{k+1}[\beta e_1, \bar{H}_k]$ computed with MGS and yielding the relation

$$\min_y \|\widetilde{b} - C_k y\|_2 = \min_y \|\beta e_1 - \bar{H}_k y\|_2, \tag{3.1}$$

where C_k and $\widetilde{b}$ are formed, respectively, at lines 1 and 2 of Algorithm 2, and where $V_{k+1} = [V_k, v_{k+1}] \in \mathbb{R}^{n\times(k+1)}$, $\bar{H}_k \in \mathbb{R}^{(k+1)\times k}$ upper Hessenberg, $e_1 = (1, 0, \ldots, 0) \in \mathbb{R}^{k+1}$, and $\beta = \|\widetilde{b}\|_2$. If we are using a left-preconditioner, $M_L \neq I$ and $M_R \equiv I$; conversely, if we are using a right- or flexible-preconditioner, $M_L \equiv I$ and $M_R \neq I$.

When computed in floating-point arithmetic, the four operations of Algorithm 2 are subject to rounding errors which affect the final attainable accuracy of the computed solution of the linear system (1.1). In this case, we note $\widehat{V}_k = [\widehat{v}_1, \ldots, \widehat{v}_k]$ to refer to the inexactly computed Krylov basis. To derive attainable error bounds for GMRES algorithms, it is often beneficial to identify a search space spanned by $\widetilde{Z}_k$ that absorbs some of these rounding errors. A natural choice for $\widetilde{Z}_k$ is the computed basis $\widehat{Z}_k = [\widehat{z}_1, \ldots, \widehat{z}_k]$, where $\widehat{z}_j = \mathrm{fl}(M_R^{-1}\widehat{v}_j)$ for $j \leq k$. However, this choice may not lead to the best attainable errors so that we keep these two quantities distinct. The difference between $\widetilde{Z}_k$ and $\widehat{Z}_k$ is technical, and only affects proofs in appendices.

Algorithm 2 Modular GMRES

1: Compute $C_k = M_L^{-1}A\widetilde{Z}_k$.
2: Compute $\widetilde{b} = M_L^{-1}b$.
3: Solve $y_k = \arg\min_y \|\widetilde{b} - C_k y\|_2$ with MGS yielding $[\widetilde{b}, C_k] = [V_k, v_{k+1}][\beta e_1, \bar{H}_k]$ and Givens rotations.
4: Compute the solution approximation $x_k = \widetilde{Z}_k y_k$.

The framework of [11] provides a generic and implementation-independent rounding error model for each operation of Algorithm 2. For this reason, it can be applied to many variations of the GMRES algorithm. For instance, the rounding error model of the matrix–matrix product at line 1 reads as follows

$$\widehat{C}_k = \mathrm{fl}\left(M_L^{-1}A\widetilde{Z}_k\right) = M_L^{-1}A\widetilde{Z}_k + \Delta_c, \qquad \|\Delta_c\|_F \leq \varepsilon_c \|M_L^{-1}A\widetilde{Z}_k\|_F, \tag{3.2}$$

where $\widehat{C}_k \in \mathbb{R}^{n\times k}$ is the result of the computed products between M_L^{-1}, A, and the computed basis $\widetilde{Z}_k$ spanning the search space. The term ε_c is a parameter bounding the magnitude of the error and describing the accuracy of the operation. Ultimately, depending on the kind of preconditioners we

use, how the products with these preconditioners are implemented, and the precisions in which these products are performed, we can obtain different levels of accuracy and, so, different values for ε_c. We emphasize that the modularity offered by the error model is crucial because it will allow us to cover a wide range of preconditioners and mixed precision implementations.

In the same vein, the model assumes that the computed preconditioned right-hand side $\widehat{b}$ at line 2 satisfies

$$\widehat{b} = \mathrm{fl}\left(M_L^{-1} b\right) = \widetilde{b} + \Delta_b, \quad \|\Delta_b\|_2 \leq \varepsilon_b \|\widetilde{b}\|_2, \tag{3.3}$$

with accuracy parameter ε_b.

The least squares problem at line 3 is solved identically for all the preconditioned GMRES implementations considered in this article. Namely, it consists of the MGS orthogonalization of $[\widetilde{b}, C_k]$ to reduce the least squares problem at line 3 to the one in (3.1), followed by Givens rotations to triangularize $\bar{H}_k$, and, lastly, the application of one triangular solve to recover the solution y_k; this is the classic implementation proposed in the GMRES founding article [32]. This process has been studied for instance in [30, sect. 7] or [11, sect. 5.3] under rounding errors, and delivers a computed solution $\widehat{y}_k$ satisfying for all $j \leq k+1$

$$\widehat{y}_k = \arg\min_y \|\widehat{b} + \Delta_{\mathrm{ls}}^b - \left(\widehat{C}_k + \Delta_{\mathrm{ls}}^c\right) y\|_2, \quad \|\left[\Delta_{\mathrm{ls}}^b, \Delta_{\mathrm{ls}}^c\right] e_j\|_2 \leq c(n,k) u_g \|\left[\widehat{b}, \widehat{C}_k\right] e_j\|_2,$$

if $\widehat{C}_k$ is not numerically singular, that is, $u_g \kappa(\widehat{C}_k) \ll 1$.

Lastly, the computed approximate solution $\widehat{x}_k$ at line 4 satisfies

$$\widehat{x}_k = \mathrm{fl}\left(\widetilde{Z}_k \widehat{y}_k\right) = \widetilde{Z}_k \widehat{y}_k + \Delta_x, \quad \|\Delta_x\|_2 \leq \varepsilon_x \|\widetilde{Z}_k \widehat{y}_k\|_2, \tag{3.4}$$

with accuracy parameter ε_x.

The original description of the rounding error models (3.2) to (3.4) can be found in [11, sect. 3.1]. Note that, for the sake of capturing a sharper forward error bound, the model (3.4) associated with the computation of the approximate solution at line 4 has been slightly changed. Moreover, contrary to [11], we do not need a rounding error model for the least squares problem at line 3. The accuracy of this process is already established since we enforce the use of MGS orthogonalization.

In addition to the previous error models, we must ensure that the remaining conditions [11, eqs. (3.5) to (3.8)] are met. Because we use specifically the MGS orthogonalization, these conditions reduce to verifying that

$$(\varepsilon_c + \varepsilon_b + u_g)\kappa(M_L^{-1} A \widetilde{Z}_k) \ll 1. \tag{3.5}$$

and

$$\varepsilon_x \ll 1. \tag{3.6}$$

The last condition (3.6) was previously [11, eq. (3.5)] which requires the basis $\widetilde{Z}_k$ not to be numerically rank deficient to the accuracy parameter ε_x. Because of the change operated in the rounding error model (3.4), the new condition (3.6) is actually less stringent.

Under the previous error models and conditions, we write Theorem 3.1 which provides an improved attainable forward error bound.

Theorem 3.1 *Suppose that Algorithm 2 is applied with a full-rank basis $\widetilde{Z}_k \in \mathbb{R}^{n\times k}$ of increasing dimension $k \leq n$ and a nonsingular left-preconditioner $M_L \in \mathbb{R}^{n\times n}$ to solve $Ax = b$. Then, there exists a*

dimension $k \leq n$ for $\widetilde{Z}_k$ at which the resulting computed Krylov basis $\widehat{V}_k$ is well-conditioned

$$\sigma_{\min}^{-1}(\widehat{V}_k) \leq 4/3 \quad \textit{and} \quad \sigma_{\max}(\widehat{V}_k) \leq 4/3, \tag{3.7}$$

and for which, if conditions (3.2) to (3.6) are met for given parameters ε_c, ε_b, and ε_x, then for any nonsingular $M_R \in \mathbb{R}^{n\times n}$, the computed solution $\widehat{x}_k$ by Algorithm 2 has a forward error satisfying

$$\frac{\|\widehat{x}_k - x\|_2}{\|x\|_2} \lesssim c(n,k)\xi\,\kappa\big(M_L^{-1}AM_R^{-1}\big), \qquad \xi = \alpha\varepsilon_c + \beta\varepsilon_b + \beta u_g + \lambda\varepsilon_x \tag{3.8}$$

with

$$\alpha = \frac{\kappa(M_R)}{\sigma_{\min}(M_R\widetilde{Z}_k)}\frac{\|M_L^{-1}A\widetilde{Z}_k\|_F}{\|M_L^{-1}AM_R^{-1}\|_F}, \qquad \lambda = 1/\kappa\big(M_L^{-1}AM_R^{-1}\big),$$
$$\beta = \max\left(1, \frac{\|M_L^{-1}A\widetilde{Z}_k\|_F}{\|M_L^{-1}AM_R^{-1}\|_F}/\sigma_{\min}(M_R\widetilde{Z}_k)\right)\kappa(M_R),$$

where $c(n,k)$ are polynomials in n and k of low degree.

Proof Proof in Appendix A. □

Remark 1. *A notable achievement of Theorem 3.1 is to show that the forward error bound (3.8) is proportional to $\kappa(M_L^{-1}AM_R^{-1})\kappa(M_R)$, where the previous state-of-the-art forward error bounds [11, eq. (3.10)] and [13, eq. (2.11)] achieve $\kappa(M_L^{-1}A)\kappa(M_R)$[1]. By using this improvement, we will be able to derive more descriptive bounds for the attainable forward errors of right- and flexible-preconditioned GMRES.*

3.2. *Error bound for left-preconditioned GMRES*

We first consider the left-preconditioned case outlined in Algorithm 1 (left) for which $M_L \neq I$ and $M_R \equiv I$. The successive matrix–vector products $M_L^{-1}A\widehat{v}_j$ for $j \leq k$ computed at lines 6 and 7 of Algorithm 1 can be compacted and rewritten under the form of the matrix–matrix product $M_L^{-1}A\widehat{V}_k$ at line 1 of Algorithm 2. We assume that these successive products are operated by the application of a standard matrix–vector product with the matrix A in precision u_a followed by the application of the preconditioner M_L^{-1} on the resulting vector in precision u_m. To study this kernel with the least amount of assumptions on the preconditioner, we assume that the application of M_L to a vector yields a computed result satisfying

$$\mathrm{fl}\left(M_L^{-1}\widehat{w}'_j\right) = \left(M_L^{-1} + \Delta M_L^{(j)}\right)\widehat{w}'_j, \qquad \|\Delta M_L^{(j)}\|_F \leq c(n,k)u_m\eta_L\|M_L^{-1}\|_F, \tag{3.9}$$

where $\Delta M_L^{(j)}$ models a generic error bounded in norm by the precision u_m and a scalar η_L, which quantifies the magnitude of the error. The value of η_L varies depending on the kind of preconditioner and how it is applied to a vector.

---

[1] In [13, eq. (2.11)], the authors actually use the term $\|\widehat{Z}_k\|_F\|M_R(\widehat{x}_k - \widehat{x}_0)\|_2/\|\widehat{x}_k\|_2$ instead of $\kappa(M_R)$, and argue that the former can be smaller, where $\widehat{x}_0$ is the initial guess. While this may be true for certain linear systems and choices of preconditioner, we have not observed it to be true in general. For this reason we prefer using $\kappa(M_R)$ in our bounds, which we find easier to read and interpret. However, a deeper investigation of the term $\|\widehat{Z}_k\|_F\|M_R(\widehat{x}_k - \widehat{x}_0)\|_2/\|\widehat{x}_k\|_2$ is certainly of interest.

Identically, we assume that the computation of the preconditioned right-hand side at line 2 of Algorithm 2 corresponds to one application of the preconditioner M_L to b satisfying (3.9). Moreover, we assume that the solution approximation $\widehat{x}_k$ obtained at line 4 is computed through a standard matrix–vector product with $\widehat{V}_k$ in precision u_g.

Lastly, we define the two following quantities to express our bound:

$$u_m \rho_M^L = \max_{j \leq k} \left(\|A^{-1} M_L \Delta M_L^{(j)} A \widehat{v}_j\|_2 \right) \tag{3.10}$$

and

$$u_m \rho_b^L = c(n,k) u_m \eta_L \frac{\|M_L^{-1}\|_F \|b\|_2}{\|M_L^{-1} b\|_2}. \tag{3.11}$$

These two quantities are used to account for the possible cancellation in the products $A\widehat{v}_j$ and $M_L^{-1}b$, and simplifications that can occur depending on the form of the error $\Delta M_L^{(j)}$.

Under this rounding error model for applying the left-preconditioner in GMRES, we provide the following theorem which bounds the attainable forward error of the computed solution.

Theorem 3.2 *Consider the left-preconditioned GMRES described by Algorithm 1 for the solution of a nonsingular linear system $Ax = b$. In addition, assume that*

$$\max \left(u_g \kappa\left(M_L^{-1}A\right), u_m \rho_b^L \kappa\left(M_L^{-1}A\right), u_m \rho_M^L, u_a \kappa(A) \right) \ll 1. \tag{3.12}$$

Then, there exists an iteration $k \leq n$ at which the computed solution $\widehat{x}_k$ has a forward error satisfying

$$\frac{\|\widehat{x}_k - x\|_2}{\|x\|_2} \lesssim c(n,k) \left(\left(u_m \rho_b^L + u_g\right) \kappa\left(M_L^{-1}A\right) + u_a \kappa(A) + u_m \rho_M^L \right). \tag{3.13}$$

Proof Proof in Appendix B. □

Remark 2. *Condition (3.12) of Theorem 3.2 is inconsequential in the sense that not meeting (3.12) implies that the right-hand side of (3.13) is of order at least 1 and, so, even if the bound (3.13) was applicable, it would not be useful. Indeed, a forward error higher than 1 implies that the computed solution has no correct digits.*

3.3. *Error bound for right-preconditioned GMRES*

We now consider the right-preconditioned case outlined in Algorithm 1 (right) for which $M_L \equiv I$ and $M_R \neq I$. Identically to the left-preconditioned case, we assume that the successive matrix–vector products $AM_R^{-1}\widehat{v}_j$ are operated by applying the matrix A in precision u_a and the preconditioner M_R^{-1} in precision u_m. In particular, we assume that the application of the right-preconditioner M_R to a vector yields

$$\mathrm{fl}\left(M_R^{-1}\widehat{v}_j\right) = \left(M_R^{-1} + \Delta M_R^{(j)}\right)\widehat{v}_j, \qquad \|\Delta M_R^{(j)}\|_F \leq c(n,k) u_m \eta_R \|M_R^{-1}\|_F, \tag{3.14}$$

where η_R is a scalar that quantifies the magnitude of the error. Moreover, we consider that the solution approximation $\widehat{x}_k$ at line 4 of Algorithm 2 is computed with a standard matrix–vector product with $\widehat{V}_k$

in precision u_g followed by the application of the right-preconditioner M_R in precision u_m. Without left-preconditioner, we have $\widetilde{b} \equiv b$ at line 2 of Algorithm 2, and no computation is needed.

Finally, we require the following quantity to express our bound; we define

$$\rho_A^R = \frac{\| |\widehat{Z}_k| |\widehat{y}_k| \|_2}{\|\widehat{Z}_k \widehat{y}_k\|_2}, \tag{3.15}$$

where $\widehat{Z}_k = [\widehat{z}_1, \ldots, \widehat{z}_k]$ and $\widehat{z}_j = \mathrm{fl}(M_R^{-1}\widehat{v}_j)$ for $j \leq k$.

With this previous rounding error model, we derive the right-preconditioned counterpart to Theorem 3.2.

Theorem 3.3 *Consider the right-preconditioned GMRES described by Algorithm 1 for the solution of a nonsingular linear system $Ax = b$. In addition, assume that*

$$\max\left(u_g\kappa(AM_R^{-1}), u_g\kappa(M_R), u_m\eta_R\kappa(M_R), u_a\kappa(A)\rho_A^R, u_a\kappa(AM_R^{-1})\kappa(M_R)\right) \ll 1. \tag{3.16}$$

Then, there exists an iteration $k \leq n$ at which the computed solution $\widehat{x}_k$ has a forward error satisfying

$$\frac{\|\widehat{x}_k - x\|_2}{\|x\|_2} \lesssim c(n,k)\left(u_g\kappa(AM_R^{-1})\kappa(M_R) + u_m\eta_R\kappa(M_R) + u_a\kappa(A)\rho_A^R\right). \tag{3.17}$$

Proof Proof in Appendix C. □

Remark 3. *Condition (3.16) of Theorem 3.3 requiring $u_a\kappa(AM_R^{-1})\kappa(M_R) \ll 1$ is likely pessimistic. It could be replaced by*

$$c(n,k)u_g\kappa(A\widetilde{Z}_k) \ll 1, \tag{3.18}$$

which would also guarantee (3.17), but involves the quantity $\widetilde{Z}_k$ instead of M_R. We can use $\Delta = A\widetilde{Z}_k - AM_R^{-1}\widehat{V}_k$ to relate $\sigma_{\min}(A\widetilde{Z}_k)$ to $\sigma_{\min}(AM_R^{-1})$, which then serves to relate $\kappa(A\widetilde{Z}_k)$ to $\kappa(AM_R^{-1})$. In the worst case scenario, the error Δ can present a specific structure that will decrease $\sigma_{\min}(A\widetilde{Z}_k)$ and bring $A\widetilde{Z}_k$ closer to rank deficiency. Namely, using (3.7), we obtain the following lower bound

$$\sigma_{\min}(A\widetilde{Z}_k) \geq 3\sigma_{\min}(AM_R^{-1})/4 - \|\Delta\|_F, \tag{3.19}$$

and we require $u_a\kappa(AM_R^{-1})\kappa(M_R) \ll 1$ to guarantee $\|\Delta\|_F \ll \sigma_{\min}(AM_R^{-1})$ in the proof of Theorem 3.3. However, because Δ is the result of the rounding errors generated by the application of the preconditioner and the matrix A, we expect it to behave like noise and not to exhibit such structure. Recent studies such as [9] even suggest that rounding errors can exhibit regularization effects that could increase the smallest singular values of the matrix $A\widetilde{Z}_k$.

3.4. *Error bound for flexible-preconditioned GMRES*

Finally, we consider the flexible-preconditioned GMRES using a fixed preconditioner $M_R^{(j)} = M_R \neq I$ for all $j \leq k$ and $M_L \equiv I$. In exact arithmetic, this algorithm is mathematically equivalent to the previously studied right-preconditioned GMRES. However, the slight implementation difference leads to critical differences for mixed precision implementations.

We assume that the computation of the successive matrix–vector products $AM_R^{-1}\widehat{v}_j$ is identical to right-preconditioned GMRES, and we use the same rounding error model (3.14) for the application of M_R^{-1}. As for the computation of the solution approximation at line 4, flexible-preconditioned GMRES is slightly, but yet critically different compared with the right-preconditioned GMRES. In the case of flexible-preconditioning, the basis $\widehat{Z}_k = [\widehat{z}_1, \ldots, \widehat{z}_k]$ with $\widehat{z}_j = \mathrm{fl}(M_R^{-1}\widehat{v}_j)$ is stored explicitly, and so the solution approximation is obtained directly from a standard matrix–vector product with $\widehat{Z}_k$ in precision u_g. Specifically, we do not reapply the preconditioner M_R^{-1} as for the right-preconditioned counterpart.

Under these changes, Theorem 3.3 for the right-preconditioned case can be adapted as follows to the flexible-preconditioned GMRES algorithm.

Theorem 3.4 *Consider the flexible-preconditioned GMRES described by Algorithm 1 for the solution of a nonsingular linear system $Ax = b$. In addition, assume that*

$$\max\left(u_g\kappa(AM_R^{-1}), u_g\kappa(M_R), u_m\eta_R\kappa(M_R), u_a\kappa(A)\rho_A^R, u_a\kappa(AM_R^{-1})\kappa(M_R)\right) \ll 1. \tag{3.20}$$

Then, there exists an iteration $k \leq n$ at which the computed solution $\widehat{x}_k$ has a forward error satisfying

$$\frac{\|\widehat{x}_k - x\|_2}{\|x\|_2} \lesssim c(n,k)\left(u_g\kappa(AM_R^{-1})\kappa(M_R) + u_a\kappa(A)\rho_A^R\right). \tag{3.21}$$

Proof Proof in Appendix D. □

Remark 4. *Comparing the previous bound (3.21) with (3.17), we can observe that without the requirement to reapply the preconditioner M_R^{-1} for computing line 4 of Algorithm 2, flexible-preconditioning can spare the term $u_m\eta_R\kappa(M_R)$ and remove the dependence on the precision u_m.*

Remark 5. *Similarly to the previous Remark 3, we expect the requirements $u_m\eta_R\kappa(M_R) \ll 1$ and $u_a\kappa(AM_R^{-1})\kappa(M_R) \ll 1$ in condition (3.20) to be likely pessimistic.*

3.5. *Summary and discussion*

For easing the discussion of our theoretical findings, we will work on simplified versions of the forward error bounds (3.13), (3.17), and (3.21). To simplify them, we use the following assumptions.

- The terms η_L and η_R defined by (3.9) and (3.14) are small of order some constants. This is the case, for instance, when M_L^{-1} and M_R^{-1} are computed and formed explicitly, such that the linear actions of the preconditioners to vectors are performed with standard matrix–vector products.
- The term ρ_b^L in the bound (3.13), which is associated with the error generated while computing the preconditioned right-hand side, is small of order some constant. While it is possible to encounter examples where $\|M_L^{-1}b\|_2 \ll \|M_L^{-1}\|_F\|b\|_2$ and so where $\rho_b^L \gg 1$, the error $u_m\rho_b^L$ can be mitigated in these cases, sometimes without much resource overheads, by applying the preconditioner in higher precision only once to form the preconditioned right-hand side.
- The term ρ_A^R in the bound (3.17) is small of order some constant. Throughout the numerical experiments reported in section 4, we have observed $\||\widehat{Z}_k||\widehat{y}_k|\|_2$ to be of the same order of magnitude as $\|\widehat{Z}_k\widehat{y}_k\|_2$ leading to $\rho_A^R = c(n,k)$. Note, however, that in many instances we obtained $\|\widehat{Z}_k\|_F\|\widehat{y}_k\|_2 \gg \||\widehat{Z}_k||\widehat{y}_k|\|_2$ which is an observation that was also made by the authors of [7]. While we will not

provide further investigation on this term in this article, we recognize that a better understanding of ρ_A^R might be of interest; in particular, we managed to generate extreme non-practical cases, not reported in this article, where ρ_A^R was of order $\kappa(M_R)$.
- The polynomials $c(n,k)$ are not descriptive and can be safely ignored.
- The conditions (3.12), (3.16), and (3.20) imposing restrictions on $\kappa(A)$, $\kappa(M)$, and $\kappa(\widetilde{A})$ for the bounds (3.13), (3.17), and (3.21) to be applicable, respectively, can be safely ignored. This is because they are either inconsequential (see remark 2) or pessimistic (see remarks 3 and 5).

Under these assumptions, the bounds on the attainable forward error for left-, right-, and flexible-preconditioned GMRES become

$$\text{(left)} \qquad u_g\kappa\big(M_L^{-1}A\big) + u_m \max\big(\rho_M^L, \kappa\big(M_L^{-1}A\big)\big) + u_a\kappa\big(A\big), \tag{3.22}$$

$$\text{(right)} \qquad u_g\kappa\big(AM_R^{-1}\big)\kappa\big(M_R\big) + u_m\kappa\big(M_R\big) + u_a\kappa\big(A\big), \tag{3.23}$$

$$\text{(flexible)} \qquad u_g\kappa\big(AM_R^{-1}\big)\kappa\big(M_R\big) + u_a\kappa\big(A\big). \tag{3.24}$$

If the bounds (3.22) to (3.24) are descriptive, they quantify the effect of the precisions u_g, u_m, and u_a on the attainable forward errors. It is critical to observe that, because the multiplicative terms in front of each of these precisions can be orders of magnitude apart, the precisions u_g, u_m, and u_a can each affect the forward errors very differently. For instance, taking the bound (3.22), the multiplicative term $\kappa(M_L^{-1}A)$ associated with u_g can be far lower than the term $\kappa(A)$ that multiplies u_a. In such a configuration, the precision u_g can be lowered to balance the two terms $u_g\kappa(M_L^{-1}A)$ and $u_a\kappa(A)$ without altering the attainable forward error. This creates opportunities for mixed precision strategies.

Another major observation is that, given one of the precisions u_g, u_m, or u_a, the multiplicative term associated with this precision can depend on the preconditioning strategy. Taking for instance the precision u_g, we have $\kappa(M_L^{-1}A)$ for the left-preconditioned GMRES bound (3.22), but we have $\kappa(AM_R^{-1})\kappa(M_R)$ for the right- and flexible-preconditioning counterparts (3.23) and (3.24). In cases where $\kappa(M_R)$ is high such that $\kappa(AM_R^{-1})\kappa(M_R) \gg \kappa(M_L^{-1}A)$, this indicates that left-preconditioned GMRES might achieve superior accuracy with respect to variations of the precision u_g, particularly when it is set lower than the other precisions.

Finally, we note that if no preconditioner is employed (that is, $M = I$), then $\kappa(\widetilde{A}) = \kappa(A)$ and $\kappa_2(M) = 1$, and the bounds (3.22) to (3.24) become $(u_a + u_g)\kappa(A)$. In this case, we emphasize that there is no room for mixed precision, and setting $u_g \neq u_a$ is not meaningful, at least according to our theory.

4. Mixed precision strategies

In this section, we use our simplified forward error bounds (3.22), (3.23), and (3.24) alongside experiments on synthetic random dense problems to establish the different meaningful mixed precision strategies for setting the three precision parameters u_g, u_m, and u_a in Algorithm 1. Each presented strategy achieves different tradeoffs between leveraging resource-efficient low precision, accuracy, robustness, and number of iterations. In the process, we unify the existing combinations of the precisions u_g, u_m, and u_a already given in the literature for left-, right-, and flexible-preconditioning under the same comprehensive analysis, and extend the range of possibilities to various new combinations. As importantly, we perform a thorough comparison of left-, right-, and flexible-preconditioning for each mixed precision strategy and highlight critical numerical differences.

4.1. *Numerical experiments on random dense matrices*

In order to validate our bounds on the attainable forward errors and to help identifying the different meaningful mixed precision strategies, we first introduce our numerical experiments on synthetic random dense problems.

4.1.1. Random dense generator

We use the following random generator to create A and an associated preconditioner M. Given a logarithmic distribution of the singular values $\{1 = \sigma_1 > \sigma_2 > \cdots > \sigma_n > 0\}$, we build A with a target condition number $\kappa_2(A)$ such that

$$A = U\,\mathrm{diag}(\sigma_1, \sigma_2, \ldots, \sigma_n)V,$$

where $U \in \mathbb{R}^{n\times n}$ and $V \in \mathbb{R}^{n\times n}$ are randomly generated orthogonal matrices. Doing so, we have $\kappa_2(A) = 1/\sigma_n$. Then we build its preconditioner M with a target condition number $\kappa_2(M) \leq \kappa_2(A)$ by truncating the previous distribution of singular values at the first j such that $1/\sigma_j > \kappa_2(M)$; the remaining singular values are replaced with σ_{j-1}:

$$M = U\,\mathrm{diag}(\sigma_1, \ldots, \sigma_{j-1}, \ldots, \sigma_{j-1})V.$$

In exact arithmetic, this setup guarantees that

$$\kappa_2\big(\widetilde{A}\big) = \frac{\sigma_{j-1}}{\sigma_n} = \frac{\kappa_2\big(A\big)}{\kappa_2\big(M\big)}. \tag{4.1}$$

Hence, with this generator, a preconditioner M reducing $\kappa_2(\widetilde{A})$ close to 1 satisfies $\kappa_2(M) \approx \kappa_2(A)$ and, conversely, a preconditioner satisfying $\kappa_2(M) \ll \kappa_2(A)$ provides $\kappa_2(\widetilde{A}) \gg 1$. This behavior is close to practice since, for practical preconditioners as well, if A is ill-conditioned and $\widetilde{A}$ is very well-conditioned, then M must be ill-conditioned.

The primary interest of the previously described random generator is that it allows us to control the quantities $\kappa(A)$, $\kappa(M)$, and $\kappa(\widetilde{A})$. It is especially useful because the forward error bounds (3.22), (3.23), and (3.24) are functions of these condition numbers. Thus, except for the ρ_M^L term, all the quantities involved in the forward error bounds are in our control, and we can derive a thorough experimental evaluation of the sharpness and descriptiveness of these bounds. It also allows us to understand the roles and effects of the conditioning of each quantity on the numerical behaviors of our mixed precision preconditioned GMRES implementations.

4.1.2. Experimental settings

We have written Julia implementations of left-, right-, and flexible-preconditioned GMRES in mixed precision as described by Algorithm 1, and we use these implementations for our experiments. We have made the source code available publicly[2].

We generate the exact solution x of the linear system (1.1) randomly with uniformly distributed entries between $[0, 1]$. The right-hand side is then obtained by computing Ax in quadruple precision to avoid introducing conflicting rounding errors. We compute the initial guess to the solution as $\widehat{x_0} = M^{-1}b$. Both A and M are generated in double precision with the generator described in section 4.1.1. The

[2] https://github.com/bvieuble/XGMRES.jl

linear action of M^{-1} to a vector is performed by decomposing M into LU triangular factors subsequently used in substitution algorithms. We compute the LU triangular factors in quadruple precision to avoid introducing conflicting rounding errors and preserving the property (4.1), and then we cast them in precision u_m.

To refer to a specific arithmetic, we will use the symbols B, S, D, and Q listed in Table 1. To refer to left-, right-, or flexible-preconditioning, we use the symbols L, R, and F, respectively. In addition, throughout the rest of the article, each variant of preconditioned GMRES will be presented in the form of a quadruplet ($\{$L-,R-,F-$\}$,u_a,u_g,u_m), so R-DSS means right-preconditioned GMRES with $u_a =$ D, $u_g =$ S, and $u_m =$ S. When we only use a triplet (u_a,u_g,u_m), such as writing DSS, it refers to the combination of precision independently of the preconditioning technique.

The attainable forward error of preconditioned GMRES in mixed precision varies according to the condition numbers of A, M, and $\widetilde{A}$, the preconditioning technique (left-, right-, and flexible-preconditioning), and the set of precisions used. In order to provide a fair experimental comparison of our different implementations of preconditioned GMRES, we employ an iterative refinement process to improve all the forward errors to the same prescribed accuracy $\|\widehat{x}-x\|_2/\|x\|_2 \leq 10^{-10}$. Hence, the methods are always compared over delivering solutions of equivalent quality. We implement iterative refinement as a restarted GMRES algorithm where the linear system residual $r_i = A\widehat{x}_i - b$ at the ith restart is computed in quadruple precision and the update of the solution $\widehat{x}_{i+1} = \widehat{x}_i + \widehat{d}_i$ where $\widehat{d}_i = \mathrm{fl}(\widehat{Z}_k\widehat{y}_k)$ is computed in double precision. Specifically, we refer to $\widehat{d}_i$ as the ith correction, and it can be viewed as the computed solution of the correction system $Ad_i = \widehat{r}_i$ by preconditioned GMRES in mixed precision with $x_0 = 0$ in Algorithm 1. The study of iterative refinement combined with GMRES is not the topic of this article. We refer readers interested in this topic to [14], [15], [4], or [11]. In particular, the link between iterative refinement and restarting is explained in [11, sect. 4]. We restart GMRES based on a tolerance τ on the preconditioned backward error

$$\frac{\|M_L^{-1}A\widehat{d}_i - M_L^{-1}r_i\|_2}{\|M_L^{-1}r_i\|_2} \leq \tau.$$

We monitor the cumulated number of inner GMRES iterations over all the restart iterations; we often refer to it simply as "the number of GMRES iterations". Because restarting more or less frequently can lead to substantially different numbers of GMRES iterations, we run mixed precision preconditioned GMRES on each problem for ten values of τ (10^{-12}, 10^{-10}, 10^{-8}, 10^{-6}, 10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}, and 5×10^{-1}), and we keep the best number to reach the prescribed accuracy $\|\widehat{x}_i - x\|_2/\|x\|_2 \leq 10^{-10}$.

4.1.3. Experimental results

Figures 1 to 6 are heatmaps of the average number of GMRES iterations over $\kappa_2(A) = 10^c$ on the x-axis and $\kappa_2(M) = 10^c$ on the y-axis for $c = 0{:}\ 16$. Each figure illustrates one mixed precision strategy that we will present in the next section 4.2. For a given pair $(\kappa_2(A), \kappa_2(M))$, the average number of iterations is computed by generating 10 random 50×50 matrices A and M of corresponding condition numbers. In particular, A and M satisfy the property (4.1) and, so, for each pair $(\kappa_2(A), \kappa_2(M))$ we have $\kappa_2(\widetilde{A}) = \kappa_2(A)/\kappa_2(M)$. Each pair is represented by a tile with a specific color tone; the lighter the color, the higher the number of iterations. If the tile is fully blank, it means that the method is unable to provide a solution with the prescribed accuracy $\|\widehat{x}_i - x\|_2/\|x\|_2 \leq 10^{-10}$. Because we are using a restarted/iterative refinement process, a blank tile indicates that mixed precision preconditioned GMRES is unable to compute corrections $\widehat{d}_i$ with at least a few correct digits, or equivalently, with a forward error reasonably lower than 1. Thus, because of the restart process, these figures do not report a

direct measurement of the attainable forward errors by Algorithm 1. Instead, through colored and blank tiles, they describe when Algorithm 1 computes solutions with forward errors lower or higher than 1.

We note that we have observed a few instances where a prescribed accuracy of order $\|\widehat{x}_i - x\|_2/\|x\|_2 \leq 10^{-10}$ was achievable but one of order 10^{-16} (i.e., double precision accuracy) was not. When this happens, we also observed that the forward errors of the computed corrections $\widehat{d}_i$ are not always lower than 1, even though the corresponding tile is colored since the prescribed accuracy is reached. As 10^{-10} is satisfactory in most practical cases, we recorded our result based on this accuracy. Nevertheless, we will mention these cases in the coming section 4.2.

Note that the upper-left triangular parts of the plots are always blank because we do not allow $\kappa_2(M) > \kappa_2(A)$. Overall, from this set of colored tiles, we define what we call the *experimental robustness* of a given variant of mixed precision preconditioned GMRES; that is, the set of triplets $(\kappa_2(A), \kappa_2(M), \kappa_2(\widetilde{A}))$ for which the (restarted) variant provides computed solutions for our synthetic problems meeting the prescribed accuracy. Finally, a white dot in the tile means that the theoretical forward error bound of a given variant, obtained from (3.22), (3.23), or (3.24), is below 1. Hence, on these white dotted tiles, the variant is guaranteed theoretically to provide a solution meeting the prescribed accuracy $\|\widehat{x}_i - x\|_2/\|x\|_2 \leq 10^{-10}$ after, if necessary, a few restarts. Particularly, for the left-preconditioned GMRES bound (3.22), we used the pessimistic assertion $u_m\rho_M^L \lesssim c(n,k)u_m\kappa(M_L^{-1}A)\kappa(M_L)$ to compute the dots. The set of white-dotted tiles defines what we call the *theoretical robustness* of the variant.

We monitor in total fourteen combinations of precisions spread over Figures 1 to 6: DDD, SSS, BBB, DSD, DBD, SBS, DDS, DDB, SSB, DSS, DBB, SBB, DSB, DBS, each run for left-, right-, and flexible-preconditioned GMRES. These combinations have been picked to be good representatives of existing and new mixed precision strategies, illustrate critical differences between left-, right-, and flexible-preconditioning, and validate our theoretical forward error bounds.

To illustrate how these heatmaps can be read, let us take variant L-SSS on the second row of Figure 1. For simplicity of exposition, we write $\kappa(\cdot)$ without subscript, but emphasize that the figures measure $\kappa_2(\cdot)$. On this plot, we observe that the variant is only able to provide correct solutions for $\kappa(A) \leq 10^8$ because the tiles corresponding to $\kappa(A) > 10^8$ are blank. In addition, we observe that the color tone is darker near the diagonal (i.e., $\kappa(A) \approx \kappa(M)$) and gets lighter below, meaning that we do fewer iterations when $\kappa(A) \approx \kappa(M)$; for example, see the tile $(\kappa(A) = 10^6, \kappa(M) = 10^6)$ compared with the tile $(\kappa(A) = 10^6, \kappa(M) = 10^0)$.

4.2. *Identifying the mixed precision strategies*

Throughout sections 4.2.1 to 4.2.6, we present six different strategies for choosing the precisions u_g, u_m, and u_a. These strategies are motivated theoretically through studying the bounds on the attainable forward errors (3.22), (3.23), and (3.24), and are then validated experimentally through the analysis of Figures 1 to 6. For conciseness, our selection of six strategies does not cover the totality of the possible combinations of u_a , u_g , and u_m, but only those satisfying $u_a \leq \min(u_g, u_m)$. Nevertheless, we provide remarks and theoretical insights for combinations satisfying $u_a \gg \min(u_g, u_m)$ in section 4.3.

We summarize in Table 2 the mixed precision strategies that we will present in sections 4.2.1 to 4.2.6. For each of these strategies, we provide the dominant term in the bounds (3.22), (3.23), and (3.24) which we derive by assuming that $\max(\kappa(\widetilde{A}), \kappa(M)) \leq \kappa(A)$. If reducing any of the three precisions worsens the bound, the strategy is considered (theoretically) meaningful and we represent it as a white cell in Table 2. Conversely, a gray cell corresponds to a strategy where the study of the

TABLE 2 *List of the mixed precision strategies presented in section 4.2 with their associated dominant term in the bounds (3.22), (3.23), and (3.24) for left-, right-, and flexible-preconditioned GMRES. We provide citations to the relevant studies when the strategy has already been proven meaningful by the literature, otherwise we write "new". White cells correspond to strategies where the study of the forward error bounds alone can conclude that the strategy is meaningful.*

	Left	Right	Flexible
$u_a = u_g = u_m$ Section 4.2.1 Figure 1	$u_m\rho_M^L + u_a\kappa(A)$ [32]	$u_g\kappa(AM_R^{-1})\kappa(M_R)$ [32]	$u_g\kappa(AM_R^{-1})\kappa(M_R)$ [31]
$u_a = u_m \ll u_g$ Section 4.2.2 Figure 2	$u_g\kappa(M_L^{-1}A) + u_m\rho_M^L$ $+u_a\kappa(A)$ [14] [15] [4] [3] [17] [29] [16]§£	$u_g\kappa(AM_R^{-1})\kappa(M_R)$ new$	$u_g\kappa(AM_R^{-1})\kappa(M_R)$ new$
$u_a = u_g \ll u_m$ Section 4.2.3 Figure 3	$u_m\max(\rho_M^L, \kappa(M_L^{-1}A))$ $+u_a\kappa(A)$ new€	$u_g\kappa(AM_R^{-1})\kappa(M_R)$ $+u_m\kappa(M_R)$ new	$u_g\kappa(AM_R^{-1})\kappa(M_R)$ [7] [25] [13] [11] [12]
$u_a \ll u_g = u_m$ Section 4.2.4 Figure 4	$u_m\max(\rho_M^L, \kappa(M_L^{-1}A))$ $+u_a\kappa(A)$ new£	$u_g\kappa(AM_R^{-1})\kappa(M_R)$ new$	$u_g\kappa(AM_R^{-1})\kappa(M_R)$ new$
$u_a \ll u_g \ll u_m$ Section 4.2.5 Figure 5	$u_m\max(\rho_M^L, \kappa(M_L^{-1}A))$ $+u_a\kappa(A)$ new€	$u_g\kappa(AM_R^{-1})\kappa(M_R)$ $+u_m\kappa(M_R)$ new$	$u_g\kappa(AM_R^{-1})\kappa(M_R)$ new$
$u_a \ll u_m \ll u_g$ Section 4.2.6 Figure 6	$u_g\kappa(M_L^{-1}A)$ $+u_m\max(\rho_M^L, \kappa(M_L^{-1}A))$ $+u_a\kappa(A)$ new	$u_g\kappa(AM_R^{-1})\kappa(M_R)$ new$	$u_g\kappa(AM_R^{-1})\kappa(M_R)$ new$

§ Previous analyses dedicated only to specific preconditioners less general than this work.

£ Unpublished analysis in [35, chap. 7] also featured in [13].

$ The bounds on the attainable forward errors alone cannot conclude that the strategy is meaningful, but experiments on synthetic and real-life problems show that it can be.

€ The bounds on the attainable forward errors and the synthetic experiments conclude that the strategy is not meaningful, but experiments on real-life problems show that it can be for some cases.

theoretical forward error bounds alone cannot conclude the meaningfulness. However, we highlight experimentally that many of these strategies can still be meaningful.

4.2.1. $u_a = u_g = u_m$

First, consider the basic case where all operations are performed in the same precision u_g. This use case is represented by the variants BBB, SSS, and DDD in Figure 1, respectively. For all three variants, the

left-, right-, and flexible-preconditioned GMRES behave almost identically with respect to the number of iterations and the experimental and theoretical robustness. Specifically, we observe that the variants can provide a solution with prescribed accuracy when the condition number of A is approximately no more than u_g^{-1}. This observation is consistent with our bounds (3.22) to (3.24) whose dominant terms are listed in the first row of Table 2 and which, under our synthetic problem property (4.1), all become $u_a\kappa(A)$. We can also see by comparing BBB to SSS and SSS to DDD that when we increase the precision, we increase the robustness of the solver with respect to $\kappa(A)$.

Hence, left-, right-, and flexible-preconditioning are all equivalent on our synthetic problems: no one preconditioning variant is better than the others in terms of robustness and number of iterations. Note, however, that the analysis of the bounds (3.22) to (3.24) might indicate a robustness superiority of the left-preconditioned GMRES for cases where $\kappa(A) \ll \kappa(\widetilde{A})\kappa(M)$. Those cases are not exhibited in Figures 1 to 6 due to the property (4.1), but seem to be corroborated in the coming section 5 on the experiments with the sparse approximate inverse preconditioner.

4.2.2. $u_a = u_m \ll u_g$

Now consider the mixed precision strategy where both the matrix A and the preconditioner M are applied with higher precision than the rest of the operations of GMRES. To our knowledge, this strategy was first proposed in [14] to improve the robustness of left-preconditioned GMRES coupled with iterative refinement. This strategy has been further studied and extended in [15] and [4]; the latter provided a first rounding error analysis of left-preconditioned GMRES for the case $u_a = u_m \leq u_g$, but dedicated to LU factors preconditioner. This mixed precision strategy has been employed in various other studies, often coupling GMRES with an iterative refinement process; see for instance [3], [13], [17], [29], or [16]. Except for the unpublished work [35, chap. 7] whose results are recalled in [13], the existing literature only provides analyses of this strategy for left-preconditioned GMRES with very specific preconditioners; for example, LU or QR-based factorization and sparse approximate inverse. The existing literature has not investigated this strategy for right- and flexible-preconditioning.

This mixed precision strategy is represented by the variants SBS, DBD, and DSD in Figure 2. Focusing on L-DSD first, increasing sufficiently the precision u_m and u_a will make the term $u_g\kappa(M_L^{-1}A)$ dominant in the bound (3.22); see second row of Table 2, and where "sufficiently" means that we satisfy $\max(u_a\kappa(A), u_m\rho_M^L) \ll u_g\kappa(M_L^{-1}A)$. Because $\kappa(A)$ is no longer limiting, we can observe in Figure 2 that convergence is achieved for tiles with $\kappa(A) > 10^8$, unlike L-SSS in Figure 1. The set of colored tiles forms a subdiagonal band that corresponds to the set of tiles for which $\kappa(M_L^{-1}A) \leq 10^8$. This is consistent with our bound (3.22), which prescribes that the term $u_g \approx 6 \times 10^{-8}$ can absorb at most $\kappa(M_L^{-1}A) = 10^7 \sim 10^8$ for the forward error to remain less than 1. The same comments can be made for L-SBS and L-DBD, for which u_g can absorb at most $\kappa(M_L^{-1}A) = 10^2 \sim 10^3$, so that the set of colored tiles forms a thinner subdiagonal band. Note also that the experimental and theoretical robustness are very close; that is, most of the tiles on which we experimentally attain the prescribed accuracy are guaranteed theoretically to attain this accuracy. For these variants, our bounds are thus descriptive.

Take now R-DSD and F-DSD. Both variants behave very similarly in terms of robustness and number of iterations. This is consistent with the bounds (3.23) and (3.24) that are both dominated by the same term $u_g\kappa(AM_R^{-1})\kappa(M_R)$ when $u_a = u_m \ll u_g$; see second row of Table 2. In our experiments, these two variants also behave very similarly to L-DSD. They provide correct solutions on almost the same tiles as L-DSD and with roughly the same number of iterations; there are some exceptions near $\kappa(A) \approx 10^{16}$ where L-DSD converges and R-DSD and F-DSD do not or need more iterations. However, comparing R-DBD and F-DBD with L-DBD, one can observe a clear robustness superiority of left-preconditioning over right- and flexible-preconditioning. Indeed, R-DSD stops converging at roughly $\kappa(A) > 10^{12}$ where, in

the meantime, L-DBD can reach $\kappa(A) = 10^{16}$. We also observe that F-DBD is less experimentally robust than R-DBD, but it is unclear how to interpret this phenomenon and its significance. Particularly, we were unable to reproduce this observation in the experiments on real-life problems featured in the coming section 5.

Our experimental observations on right- and flexible-preconditioning are more optimistic than what the theory prescribed. Indeed, under property (4.1), we have $\kappa_2(AM_R^{-1})\kappa_2(M_R) \approx \kappa_2(A)$ such that, with $u_g = \text{S}$, the bounds (3.23) and (3.24) can only guarantee the convergence for the tiles $\kappa(A) \leq 1.7 \times 10^7$. Surprisingly, we observed on logs not reported in this document that many tiles on which R-DSD and F-DSD successfully converge to the prescribed accuracy of order 10^{-10} for $\kappa(A) > 10^8$ cannot achieve a prescribed accuracy of order 10^{-16}. From the iterative refinement theory, it means that R-DSD and F-DSD do not always compute corrections with forward errors reasonably lower than 1. This suggests that our forward error bounds (3.23) and (3.24) are not necessarily pessimistic for these cases. Instead, we believe that some numerical mechanisms, that we do not fully understand, allow the first few corrections to be computed accurately until a certain point is reached where the corrections become inaccurate; that is, without correct digits.

Overall, we showed that setting $u_a = u_m \ll u_g$ in left-preconditioned GMRES can be a valid strategy to leverage tradeoffs between, for instance, L-SSS and L-DDD (equivalently between L-BBB and L-SSS or L-BBB and L-DDD). Namely, L-DSD achieves better numerical properties than L-SSS but worse than L-DDD. The strategy can be particularly effective if the computational bottleneck is on the orthogonalization process and the storage of the Krylov basis. For this approach to be meaningful numerically, the theory prescribes that we need to be in a configuration where $\min(\kappa(A), \rho_M^L) \gg \kappa(M_L^{-1}A)$. We also conclude that, while not covered by our theory, this strategy can be successful for right- and flexible-preconditioned GMRES. Nevertheless, we recommend using the left-preconditioned variant which exhibits better theoretical and experimental robustness.

4.2.3. $u_a = u_g \ll u_m$

One can improve the solver's performance by applying the preconditioner with lower precision than the rest of the operations; that is, $u_a = u_g \ll u_m$. This strategy has been first proposed in [7] for flexible-preconditioned GMRES using LU factors preconditioner computed and applied in low precision. In this configuration, the LU factorization done once before the GMRES iterations is generally the most expensive part of the algorithm. Hence, computing the factorization in a low precision $u_m \gg u_g$ and keeping the resulting factors in low precision in memory can achieve substantial resource savings. Note that a large part of the theoretical analysis of [7] is not specific to LU factors preconditioner, but is generic and covers a wide variety of preconditioners. Flexible-preconditioned GMRES using low precision LU factors as preconditioner has been implemented in the HSL_MA79 sparse solver [25]. Another proof of the results of [7] is provided in [11]. This mixed precision strategy has also been further extended in [13] and the subsequent associated study [12] to split-preconditioned flexible GMRES. Except for early comments in the unpublished work [35, chap. 7] for the left-preconditioned case that are further substantiated in [13], the existing literature did not investigate this strategy for left- and right-preconditioned GMRES and focused solely on flexible-preconditioned GMRES.

We use the variants SSB, DDB, and DDS in Figure 3 to illustrate this mixed precision strategy. We comment on the left-preconditioned case first. The theory supports the idea that switching from $u_a = u_m = u_g$ to $u_a = u_g \ll u_m$ by decreasing the precision u_m will increase the bound (3.22). The dominant term then becomes $u_m \max(\rho_M^L, \kappa(M_L^{-1}A))$ if u_a is small enough compared with u_m (third row of Table 2), and is greater than the dominant term $u_a\kappa(A)$ for $u_a = u_m = u_g$ (first row of Table 2). This

translates effectively into a robustness decrease in our experiments. Indeed, comparing L-DDD and L-DDS where u_m is reduced from D to S, we observe that L-DDS converges on significantly fewer tiles than L-DDD in Figure 1. On the other hand, L-DDS presents a better experimental robustness than L-SSS and, therefore, achieves a tradeoff between L-DDD and L-SSS; the same conclusion applies to L-DDB and L-SSB. We will develop in more details on what makes this tradeoff possible in the next section 4.2.4, where we also explain that a mixed precision strategy using strictly lower precision can achieve the same robustness as L-SSB, L-DDB, or L-DDS and should therefore be preferred. We recall that, because it is inconvenient to compute ρ_M^L, the white dots in Figure 3 and other figures have been obtained using the pessimistic assertion $u_m\rho_M^L \lesssim c(n,k)u_m\kappa(M_L^{-1}A)\kappa(M_L)$.

The conclusion is very different for the flexible-preconditioned GMRES case. Indeed, one can observe that the associated forward error bound (3.24) is independent of the precision u_m in which the preconditioner is applied. In other words, switching from $u_a = u_m = u_g$ to $u_a = u_g \ll u_m$ by decreasing the precision u_m leaves the dominant term $u_g\kappa(AM_R^{-1})\kappa(M_R)$ unaffected in the bound (3.24); compare the first and third row of Table 2. Hence, regardless of how low the precision u_m is, the attainable forward error of flexible-preconditioned GMRES remains identical. This is a critically desirable property. For instance, comparing F-DDD with F-DDS, we see that both variants exhibit the same experimental robustness; that is, they compute correct solutions with prescribed accuracy on the same tiles. Reducing the precision u_m further from S to B by taking the variant F-DDB, we still achieve the same experimental robustness. Note also that as the white dots for F-DDS and F-DDB overlap perfectly with the colored tiles, our theoretical bound (3.24) is very descriptive for this use case. However, it should be remarked that while reducing the precision does not impact the experimental robustness, it significantly increases the number of iterations. The same comments apply when comparing F-SSB to F-SSS.

Finally, we discuss the right-preconditioned variants R-SSB, R-DDB, and R-DDS. Compared with the flexible-preconditioning bound (3.24), the right-preconditioned GMRES forward error bound (3.23) exhibits a dependence on the precision u_m. The bound (3.23) has an additional term $u_m\kappa(M_R)$ which becomes the dominant term if $u_a = u_g \ll u_m$ such that $u_g\kappa\big(AM_R^{-1}\big)\kappa\big(M_R\big) \ll u_m\kappa(M_R)$; see the third row of Table 2. Ultimately, contrary to flexible-preconditioned GMRES, setting the precision u_m low while having an ill-conditioned preconditioner will worsen the forward error of right-preconditioned GMRES. We recover in some sense the conclusion of [8] on the robustness superiority of flexible-preconditioning over right-preconditioning. Our theory translates into the experiments. Comparing R-DDB to R-DDD, we see that decreasing the precision u_m from D to B reduces the robustness theoretically and experimentally. Moreover, the effect of the dominant term $u_m\kappa(M_R)$ on the experimental robustness of R-DDB can be observed very distinctly: it limits the convergence to the horizontal band of tiles satisfying $\kappa(M_R) \leq 10^4$. We also observe that increasing the precision u_m in R-DDB from B to S increases the robustness. This is justified by the theory since the term $u_m\kappa(M_R)$ will be reduced, and justified by our experiments by comparing R-DDB and R-DDS. When compared with flexible-preconditioning, it is worth pointing out that while R-DDB is less robust than F-DDB, F-DDB has to store two bases and, for an identical number of iterations, requires double the amount of memory. Moreover, right-preconditioning offers a more controlled loss of robustness when decreasing the precision u_m than left-preconditioning. Because of the complex form of ρ_M^L, it is not convenient to compare the theoretical dominant terms $u_m\rho_M^L$ (left-preconditioning) and $u_m\kappa(M_R)$ (right-preconditioning) directly. Nonetheless, the experiments are strikingly clear: R-SSB, R-DDB, and R-DDS outperform their left-preconditioned counterparts. Lastly, to comment specifically on R-SSB, we see that this variant exhibits both the limitations of having $u_a = u_g =$ S which requires $\kappa(A) \leq 1.7 \times 10^7$, and having $u_m =$ B which requires $\kappa(M) \leq 2.6 \times 10^2$ similarily to R-DDB.

Setting $u_a = u_g \ll u_m$ can lead to substantial resource savings if the application of the preconditioner is the computational bottleneck of the algorithm. We recover the conclusion of the existing literature stating that the attainable forward error of flexible-preconditioning is insensitive to the choice of precision u_m, and is therefore particularly attractive for this strategy. On the other hand, we found this strategy to be suitable for right-preconditioning. While it would not offer the same level of robustness as flexible-preconditioned GMRES, it presents a more controlled loss of robustness than left-preconditioning, and it may achieve a better memory footprint than flexible-preconditioning.

4.2.4. $u_a \ll u_g = u_m$

The previous strategy $u_a = u_m \ll u_g$ described in section 4.2.2 makes it possible to compute the orthogonalization and store the Krylov basis in lower precision. On the other hand, the strategy $u_a = u_g \ll u_m$ developed in section 4.2.3 allows the preconditioner to be applied and stored in a lower precision. Yet, there are instances where both the orthogonalization and the preconditioner can be costly. In the following, we demonstrate that applying only A in high precision while having $u_m = u_g$ in lower precision can preserve some of the numerical benefits of the previously presented mixed precision strategies. This approach appeared in the unpublished work [35, chap. 7] and has been subsequently employed in [13] and in the follow-up study [12].

This mixed precision strategy is represented by the variants SBB, DBB, and DSS in Figure 4. Focusing first on left-preconditioning, the bound (3.22) tells us that the forward error will be driven by the term $u_m \max(\rho_M^L, \kappa(M_L^{-1}A)) + u_a\kappa(A)$; see the fourth row of Table 2. Compared with a combination of precision $u_a = u_g = u_m$ whose forward error is driven by the term $u_a\kappa(A)$ if $\rho_M^L \leq \kappa(A)$, increasing the precision u_a alone to obtain $u_a \ll u_g = u_m$ will improve the forward error if $\max(\rho_M^L, \kappa(M_L^{-1}A)) \ll \kappa(A)$. Since under property (4.1) we have $\kappa(M_L^{-1}A) \ll \kappa(A)$ for $\kappa(M) \gg 1$, we need more specifically $\rho_M^L \ll \kappa(A)$ to improve the forward error bound. From the definition of ρ_M^L in (3.10), we have

$$\frac{\rho_M^L}{\kappa(A)} \leq \kappa(M_L^{-1}A)\frac{\max_j \|A\widehat{v}_j\|_2}{\|A\|_F}.$$

Remarkably, due to the form of the Krylov basis vectors $\widehat{v}_j$, the product $A\widehat{v}_j$ can exhibit high cancellation such that $\|A\widehat{v}_j\|_2/\|A\|_F \ll 1$ leading potentially to a very small ratio $\rho_M^L/\kappa(A) \ll 1$. We will not attempt to justify this property theoretically in this article. Instead we report in Figure 7 the evolution of $\|A\widehat{v}_j\|_2$ and $\||A|\|\widehat{v}_j|\|_2 \lesssim \|A\|_F$ throughout the GMRES iterations for $\kappa(A) = 10^{16}$ fixed and increasing $\kappa(M_L)$. We can observe that, as $\kappa(M_L)$ increases, the ratio $\max_j \|A\widehat{v}_j\|_2/\|A\|_F$ decreases, to the point where, when $\kappa(M_L) = 10^{16}$, we almost reach $\rho_M^L/\kappa(A) \approx 10^{-16}$. Hence, when $\kappa(M_L)$ gets closer to $\kappa(A) \gg \kappa(M_L^{-1}A)$ and $\kappa_2(M_L^{-1}A)$ gets closer to 1, it becomes meaningful to set $u_a \ll u_m$. We note that the high cancellation in the product $A\widehat{v}_j$ making this strategy possible is preconditioner dependent and is not always observable for certain classes of preconditioners.

On the other hand, the theory also supports the idea that having $u_g \ll u_m$ is not meaningful for left-preconditioning. Indeed, whether $u_a = u_g \ll u_m$ or $u_a \ll u_g = u_m$, in both cases the dominant term in (3.22) is $u_m \max(\rho_M^L, \kappa(M_L^{-1}A)) + u_a\kappa(A)$ so that keeping $u_g \ll u_m$ does not provide any theoretical improvement. Thus, the previous strategy $u_a = u_g \ll u_m$ presented in section 4.2.3 is not theoretically meaningful with left-preconditioning, and the strategy $u_a \ll u_g = u_m$ currently considered should be preferred. Note that we will slightly mitigate this claim in section 5 when presenting experiments on real-life problems and where we expose cases where $u_a = u_g \ll u_m$ can still offer advantageous tradeoffs with left-preconditioning.

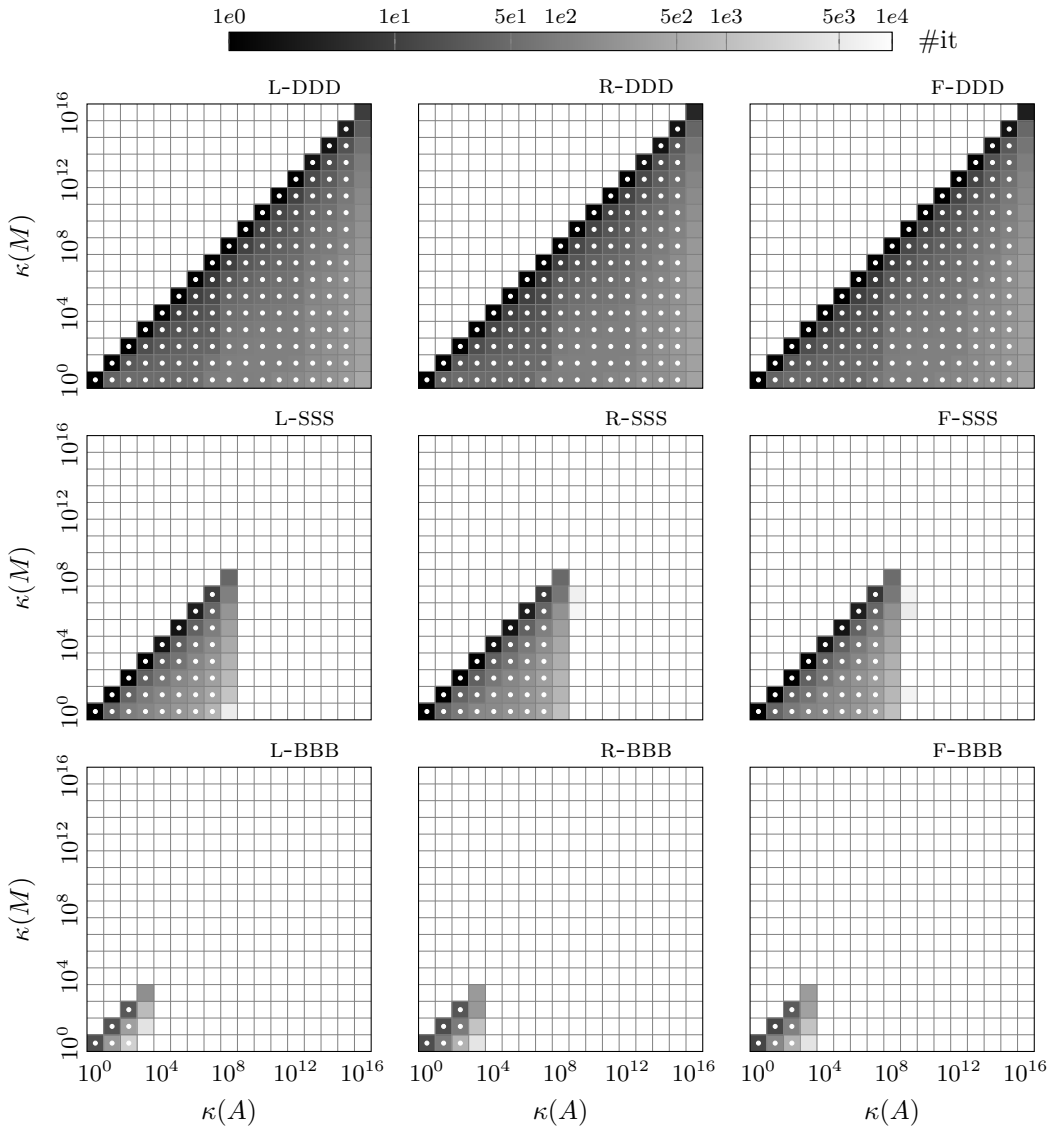

FIG. 1. Average number of iterations for mixed precision preconditioned (restarted) GMRES according to $\kappa_2(A)$ and $\kappa_2(M)$ for $u_a = u_g = u_m$ and for left-, right- and flexible-preconditioning. The iterations are stopped when the algorithm converges to $\|x-\hat{x}\|_2/\|x\|_2 \leq 1\times 10^{-10}$. A white dot in the tile means that our theory guarantees the convergence.

The theory appears consistent with the numerical behavior of the variant L-DSS in Figure 4. While the variant L-DSS is less robust than L-DSD in Figure 2, correct solutions can still be achieved for tiles where $\kappa(A) > 10^8$ compared with L-SSS in Figure 1, making L-DSS a tradeoff between L-DSD and L-SSS. If we now compare L-DSS to L-DDS in Figure 3, we can see that L-DSS achieves equivalent robustness while using strictly lower precision than L-DDS. This observation supports the idea that

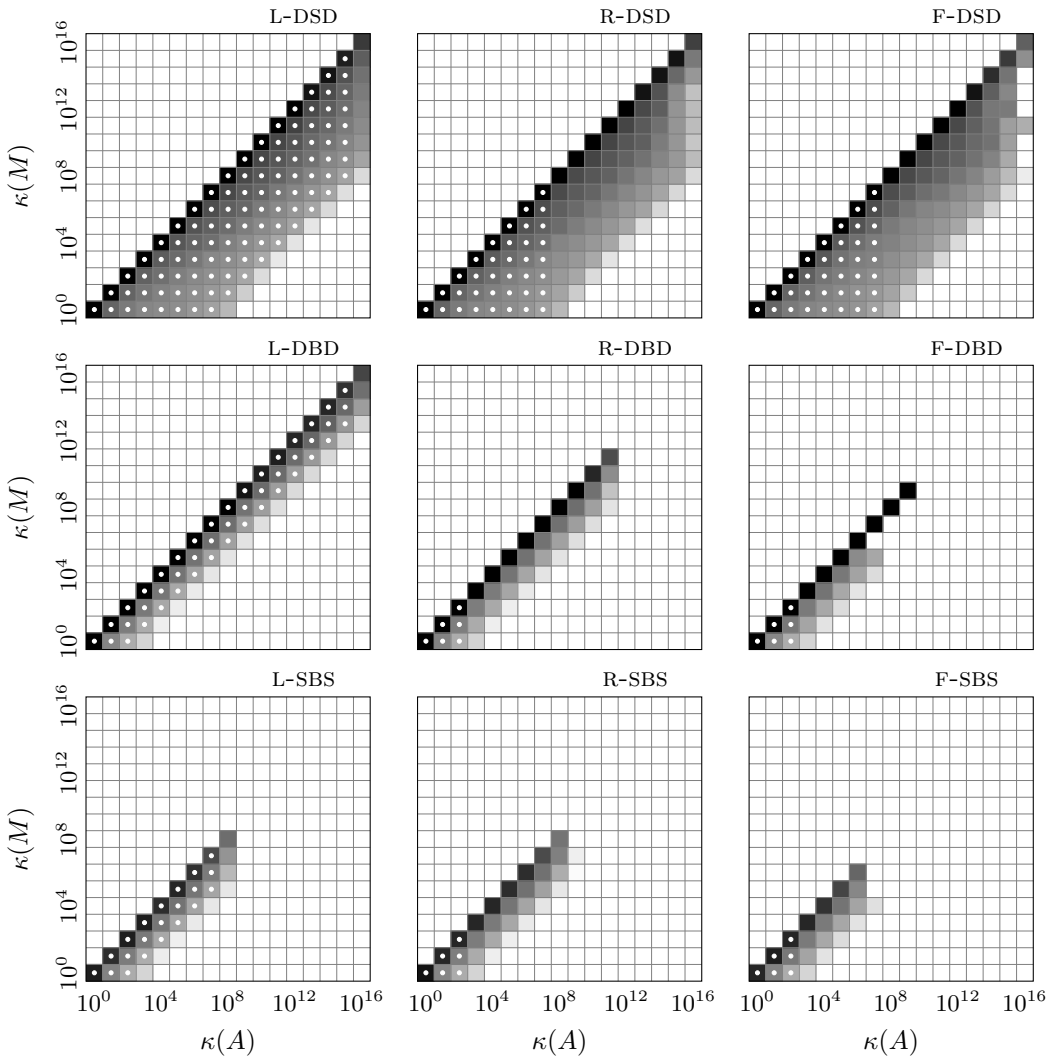

FIG. 2. Same as Figure 1 but for $u_a = u_m \ll u_g$.

$u_a = u_g \ll u_m$ is not meaningful with left-preconditioning. Similar comments can be made for the variant L-SBB. Regarding the variant DBB, since it achieves the same numerical robustness as SBB but involves strictly higher precision, it is therefore numerically not meaningful.

This strategy can also be successful for right- and flexible-preconditioning. Taking the variants R-DSS and F-DSS, we can observe that they are remarkably almost as robust as R-DSD and F-DSD, and more robust than the left-preconditioned variant L-DSS. Note, however, that the number of iterations is increased compared with R-DSD and F-DSD. For the same reasons as for the variants R-DSD and F-DSD discussed in section 4.2.2, the convergence of these methods for tiles verifying $\kappa(A) > 1.7 \times 10^7$ is not covered by our theory. In particular, identically to those previous variants, we observed that many

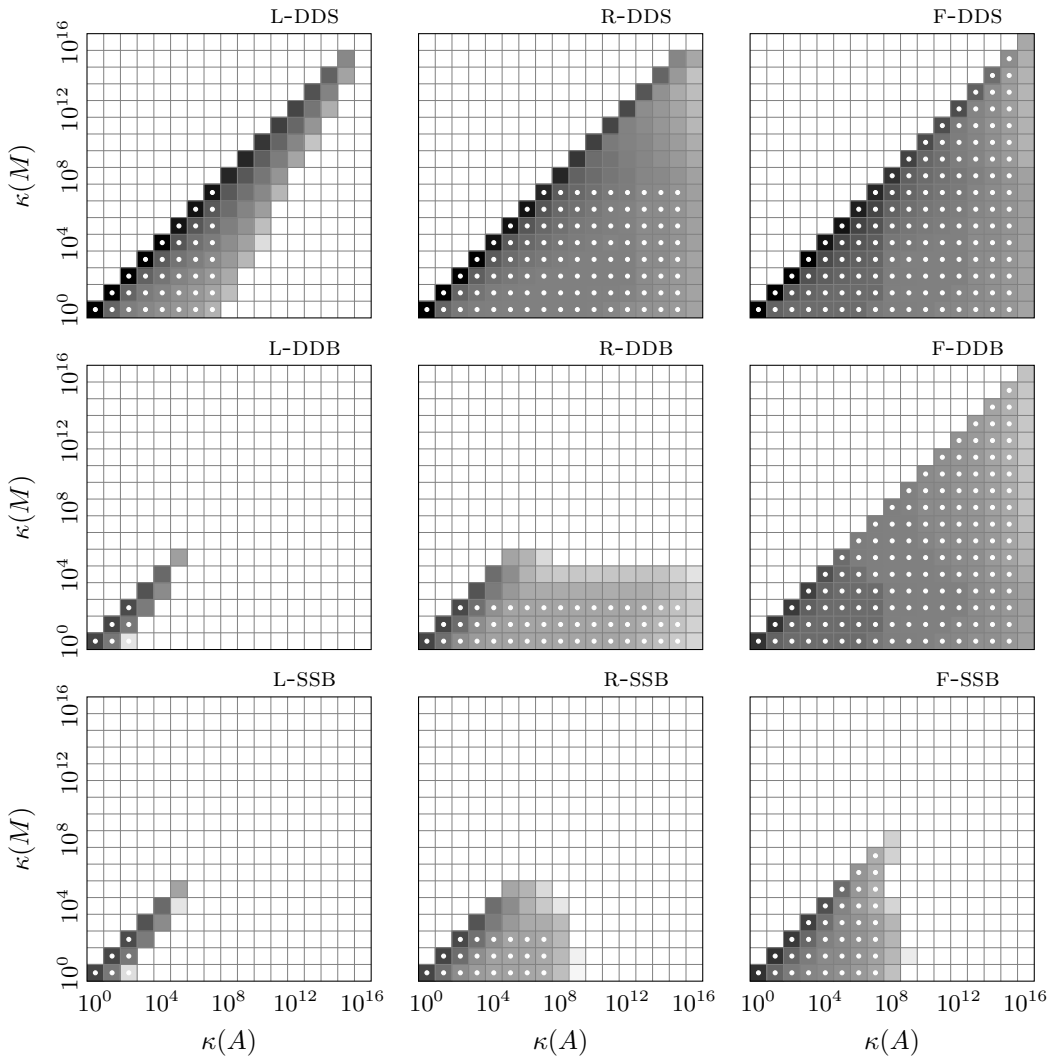

FIG. 3. Same as Figure 1 but for $u_a = u_g \ll u_m$.

tiles on which R-DSS and F-DSS successfully converge to the prescribed accuracy of order 10^{-10} for $\kappa(A) > 1.7 \times 10^7$ cannot achieve a prescribed accuracy of order 10^{-16}. For those tiles, the computed corrections $\widehat{d_i}$ at each restart do not always achieve a forward error lower than 1.

Hence, using DSS over DSD or DDS can leverage lower precision in potentially critical parts of the computation while still exposing better numerical properties than SSS. The same can be said for SBB compared with SBS and SSB. The presented strategy $u_a \ll u_g = u_m$ can be effective if both the orthogonalization process and the application of the preconditioner are costly; or if only the application of A can be performed with high precision given a hardware and software setup. We would cautiously recommend preferring right- and flexible-preconditioning in this case, which seem to perform better

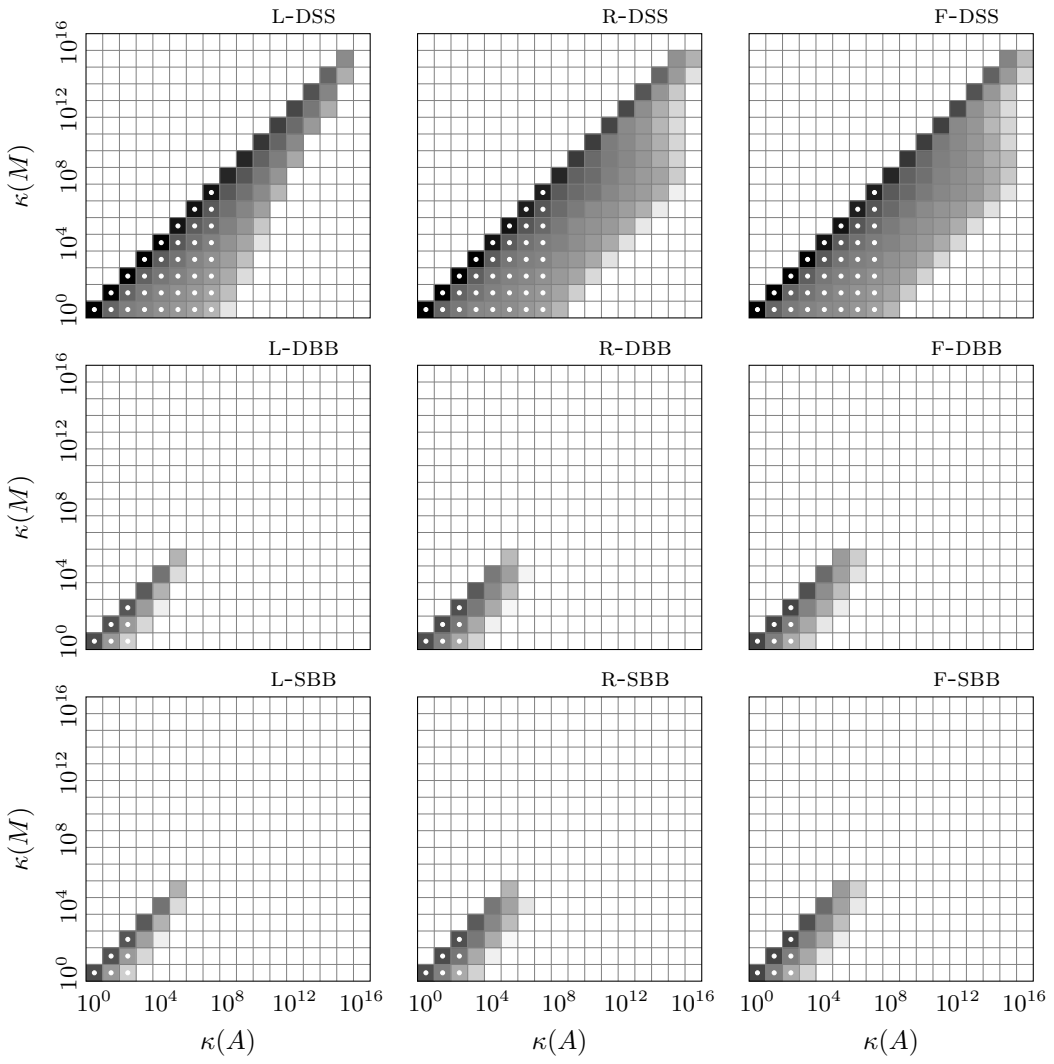

FIG. 4. Same as Figure 1 but for $u_a \ll u_g = u_m$.

on our set of synthetic problems. However, without strong theoretical guarantees, it might be the case that the left-preconditioned variant might outperform right- and flexible- on certain problems and preconditioners.

4.2.5. $u_a \ll u_g \ll u_m$

The previous strategy proposed to apply only the matrix A in a higher precision and have $u_g = u_m$ in a lower precision. We now show that there are instances where it is possible to apply the preconditioner in an even lower precision, leading to $u_a \ll u_g \ll u_m$. This proposed setup can be interesting if the storage and application of the preconditioner is the main computational bottleneck, but the orthogonalization

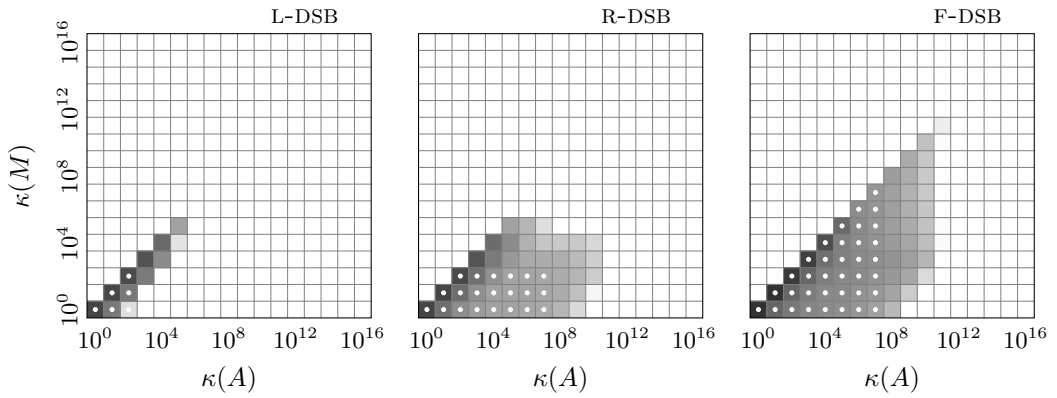

FIG. 5. Same as Figure 1 but for $u_a \ll u_g \ll u_m$.

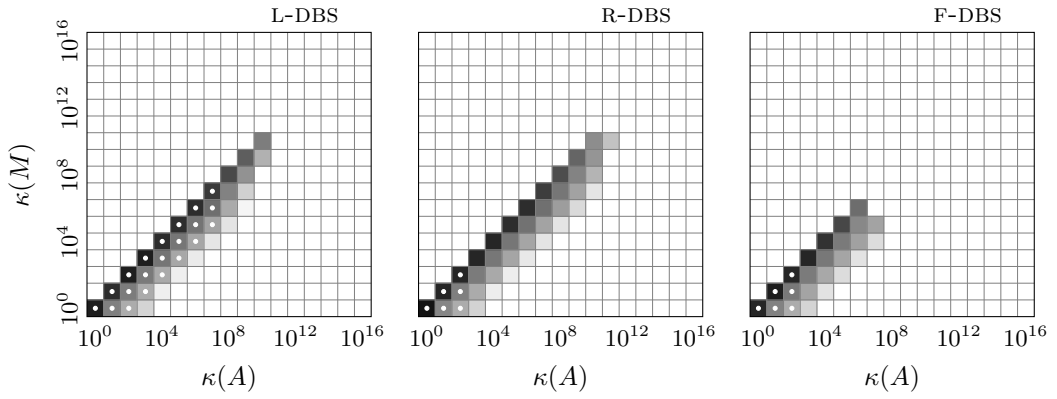

FIG. 6. Same as Figure 1 but for $u_a \ll u_m \ll u_g$.

and storage of the Krylov basis is still costly and should not be operated in high precision. Such combinations of precisions have not yet been investigated in the literature.

We represent this strategy in Figure 5 with the variant DSB. As explained in section 4.2.4, setting $u_m \gg u_g$ with left-preconditioning is not meaningful and, so, the variant L-DSB has no significant numerical benefits over L-DBB in Figure 4. On the other hand, F-DSB converges on more tiles than F-DBB in Figure 4 and F-SSB in Figure 3 and, therefore, presents a meaningful tradeoff. More specifically, F-DSB displays both: the resilience of setting $u_m \gg u_g$ for flexible-preconditioning assessed in section 4.2.3, and the increased experimental robustness of setting $u_a \ll u_g$ observed in sections 4.2.2 and 4.2.4. When comparing F-DSB with F-DSS, we also remark that reducing u_m from S to B harms noticeably the experimental robustness, so that the effect of using high precision for u_a is counterbalanced by the use of low precison for u_m. Regarding right-preconditioning, R-DSB inherits the decreased robustness over low precision u_m of variant R-SSB in Figure 3 when compared with F-SSB. For this reason, it is expected theoretically and observed experimentally that R-DSB is less robust than F-DSB. Comparing now R-DSB with R-SSB, we can observe that increasing the precision u_a from S to D presents a small but noticeable improvement on the experimental robustness: R-DSB achieves

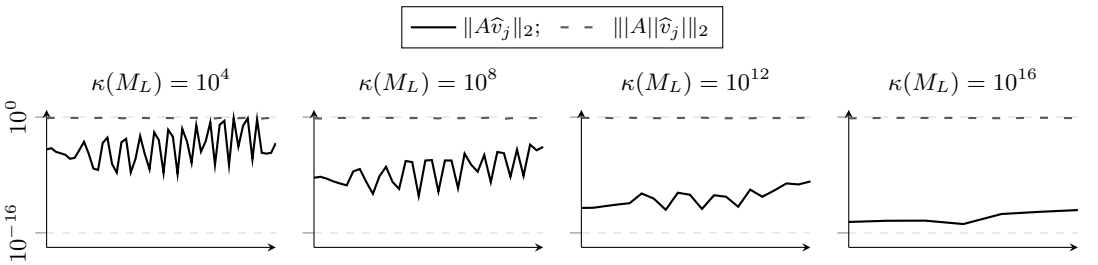

FIG. 7. Evolution of $\|A\widehat{v}_j\|_2$ and $\||A|\widehat{v}_j\|_2$ over the iterations of left-preconditioned GMRES in full double precision (i.e., $u_g = u_m = u_a =$ D). The matrix A and the preconditioners M_L are built with the random generator described in section 4.1.1. We fix $\kappa_2(A) = 10^{16}$ and make $\kappa_2(M_L)$ vary.

convergence on a few more tiles than R-SSB. As for the other right-preconditioned variants presented in sections 4.2.2 and 4.2.4 and using $u_a \ll u_g$, this is a behavior that we cannot explain by our forward error bound alone.

Overall, this strategy can be advantageous with flexible-preconditioning in cases where the main resource bottleneck is on the preconditioner but the orthogonalization and the storage of the Krylov basis are still costly. The right-preconditioned variant has a more limited range of usefulness, but can potentially be of interest on specific problems and configurations. Our rounding error analysis and synthetic problem experiments tend to indicate that left-preconditioning is not meaningful and should not be considered for this strategy.

4.2.6. $u_a \ll u_m \ll u_g$

Lastly, as for the previous strategy $u_a \ll u_g \ll u_m$ presented in section 4.2.5, we consider a strategy where all precisions u_a, u_g, and u_m are set differently. However, contrary to this previous strategy, we now show that having u_g as the lowest precision such that $u_a \ll u_m \ll u_g$ can be numerically meaningful. Such a strategy can be practical if the computational bottleneck is on the orthogonalization process and the storage of the Krylov basis, but the application of the preconditioner is still costly and should not be performed in high precision. As for the previous strategy, such combinations of precisions have not been considered in the literature.

We display in Figure 6 the variant DBS which is an implementation of this strategy. Commenting on left-preconditioning first, we can see that L-DBS presents a robustness tradeoff between L-SBS and L-DBD in Figure 2. That is, it is less robust than L-DBD but uses strictly lower precision. On the other hand, it is slightly more robust than L-SBS since it converges on a few additional tiles satisfying $\kappa(A) > 10^8$, but uses higher precision. This behavior is explained by the bound (3.22), indicating that for this setup of precisions the forward error is driven by $u_g\kappa(M_L^{-1}A) + u_m \max(\rho_M^L, \kappa(M_L^{-1}A))$ when u_a is chosen such that $u_a\kappa(A)$ is sufficiently small; see the sixth row of Table 2. The first term $u_g\kappa(M_L^{-1}A)$ imposes $\kappa(M_L^{-1}A) \leq 2.6 \times 10^2$, which restricts the convergence to the subdiagonal band as for L-SBS (and L-DBD). However, contrary to L-SBS for which the term $u_a\kappa(A)$ prevents convergence for tiles verifying $\kappa(A) > 10^8$, L-DBS is limited by the term $u_m\rho_M^L$ instead. As explained in section 4.2.4, because ρ_M^L can be significantly lower than $\kappa(A)$, L-DBS can converge on tiles satisfying $\kappa(A) > 10^8$ up to a certain limit. By increasing the precision u_m to D, L-DBD removes the limitation imposed by the term $u_m\rho_M^L$ in L-DBS, and achieves a better robustness. Lastly, because L-DBS converges on strictly more tiles than

TABLE 3 *List of the mixed precision strategies satisfying* $u_a \gg \min(u_g, u_m)$ *with their associated dominant term in the bounds* (3.22), (3.23), *and* (3.24) *for left-, right-, and flexible-preconditioned GMRES. White cells correspond to strategies where the study of the forward error bounds alone can conclude that the strategy is meaningful.*

	Left	Right	Flexible
$u_g \ll u_a = u_m$ $u_g \ll u_a \ll u_m$ Section 4.3.1	$u_m \max\left(\rho_M^L, \kappa\left(M_L^{-1}A\right)\right)$ $+u_a\kappa(A)$	$u_g\kappa\left(AM_R^{-1}\right)\kappa(M_R)$ $+u_m\kappa(M_R)$ $+u_a\kappa(A)$	$u_g\kappa\left(AM_R^{-1}\right)\kappa(M_R)$ $+u_a\kappa(A)$
$u_m \ll u_a = u_g$ $u_m \ll u_a \ll u_g$ Section 4.3.2	$u_g\kappa\left(M_L^{-1}A\right) + u_m\rho_M^L$ $+u_a\kappa(A)$	$u_g\kappa\left(AM_R^{-1}\right)\kappa(M_R)$	$u_g\kappa\left(AM_R^{-1}\right)\kappa(M_R)$
$u_a \gg \max(u_g, u_m)$ Section 4.3.3	$u_m\rho_M^L + u_a\kappa(A)$	$u_g\kappa\left(AM_R^{-1}\right)\kappa(M_R)$ $+u_a\kappa(A)$	$u_g\kappa\left(AM_R^{-1}\right)\kappa(M_R)$ $+u_a\kappa(A)$

L-DBB in Figure 4, L-DBS is more robust than all the variants using strictly lower precision and is therefore meaningful.

Similarly to the variants R-DBB and L-DBB in section 4.2.4, and most likely for the same reasons, R-DBS presents a slightly improved experimental robustness than L-DBS. As for R-DBB, the convergence of R-DBS is not explained by our theory alone for tiles satisfying $\kappa(A) > 2.6 \times 10^2$. Taking now flexible-preconditioning, we observe that F-DBS is less experimentally robust than R-DBS. We made a similar comment when comparing F-DBD and R-DBD in section 4.2.2. Identically to this previous case, we are not sure how to interpret the significance of this observation.

To conclude, the new variant DBS offers a robustness tradeoff between SBS and DBD. Because of the narrow range of improvement over SBS, we imagine that the practical configurations in which this variant is useful is limited. However, given the right problem, our theory and numerical experiments suggest that it can be successful for left- and right-preconditioning.

4.3. *Comments about the case* $u_a \gg \min(u_g, u_m)$

The six previous strategies presented in sections 4.2.1 to 4.2.6 covers all the combinations satisfying $u_a \leq \min(u_g, u_m)$. For the sake of completness, we now comment on the combinations for which we have $u_a \gg \min(u_g, u_m)$, but do not validate them with numerical experiments. We emphasize that a deeper experimental investigation is necessary to assess the meaningfulness of each of these strategies, but this is something that, as we will explain, we cannot add concisely to this article. We summarize these mixed precision strategies in Table 3 which we construct by assuming that $\max(\kappa(\widetilde{A}), \kappa(M)) \leq \kappa(A)$.

4.3.1. $u_g \ll u_a = u_m$ and $u_g \ll u_a \ll u_m$

From the study of the bounds (3.23) and (3.24), the theory supports the idea that there are setups in which having $u_a \gg u_g$ can be meaningful for right- and flexible-preconditioning. For instance, practical preconditioners may not satisfy the synthetic problem property (4.1), which can lead to $\kappa(A) \ll \kappa(AM_R^{-1})\kappa(M_R)$. In this case, our theoretical bounds indicate that we can set $u_a \gg u_g$ without altering the attainable forward error of right- and flexible-preconditioned GMRES.

Because we enforce property (4.1) in our synthetic experiments, we cannot investigate their meaningfulness experimentally.

4.3.2. $u_m \ll u_a = u_g$ and $u_m \ll u_a \ll u_g$

Equivalently to section 4.3.1, by studying the bound (3.22) for left-preconditioned GMRES we conclude that if the upper bound $u_m \rho_M^L \lesssim c(n,k) u_m \kappa(M_L^{-1}A)\kappa(M_L)$ is attained and $\kappa(A) \ll \kappa(M_L^{-1}A)\kappa(M_L)$, we may set $u_a \gg u_m$.

For the same reason as for section 4.3.1, we do not investigate these strategies experimentally.

4.3.3. $u_a \gg \max(u_g, u_m)$

Setting $u_a \gg u_g$ for left-preconditioning or $u_a \gg u_m$ for right-preconditioning is, however, less likely to be numerically meaningful because it would require $\kappa(\widetilde{A}) \gg \kappa(A)$ and $\kappa(M) \gg \kappa(A)$, respectively. Indeed, we often expect preconditioning to reduce the condition number of A so that $\kappa(\widetilde{A}) \ll \kappa(A)$. In addition, because M^{-1} is constructed as an approximation of A^{-1}, we often have $\kappa(M) \leq \kappa(A)$.

Yet, accounting for the varieties of problems and preconditioning approaches to solve them, cases where $\kappa(\widetilde{A}) \gg \kappa(A)$ or $\kappa(M) \gg \kappa(A)$, for which setting $u_a \gg u_g$ (left-preconditioning) or $u_a \gg u_m$ (right-preconditioning) would be meaningful, cannot be excluded in principle. However, since we enforce $\max(\kappa(\widetilde{A}), \kappa(M)) \leq \kappa(A)$ in our synthetic experiments, we also do not provide further investigations on these strategies.

4.4. *Other discussions*

When considering which mixed precision strategy to choose, it is very important to consider that every strategy is limited to specific range of condition numbers $\kappa(A)$, $\kappa(M)$, and $\kappa(\widetilde{A})$. It means that a given strategy is not adequate or optimal for every problem.

We also emphasize that the presented strategies are mainly based on a rounding error analysis perspective but do not further consider hardware-specific constraints and properties. Hence, within the selection of possible mixed precision strategies, a user should additionally consider hardware and software features to further identify the best implementations of Algorithm 1 for their specific computing environment. It should also be clear that the practical relevance of a given strategy depends on the problem and preconditioner properties. Indeed, the best mixed precision implementation for a problem with its workload bottleneck on the orthogonalization and the storage of the Krylov basis is not the same as the best implementation for a problem where the bottleneck is on the computation and application of a costly preconditioner.

Lastly, while the simplifications operated to derive the bounds (3.22) to (3.24) allow for a clearer and intuitive interpretation of the interactions between the precisions, the conditioning of A, M, and $\widetilde{A}$, and the final attainable forward error, they may not fully describe certain practical preconditioners. Of course, one can always fall back to the more general bounds (3.13), (3.17), and (3.21) in such a situation to get more accurate theoretical insights. In the next section, we address real-life problems and practical preconditioners to evaluate and compare the benefits of these strategies on practical setups.

5. Numerical experiments on real-life SuiteSparse matrices

We now use the fourteen mixed precision variants presented in section 4, and that illustrate the six mixed precision strategies of Table 2, to solve real-life problems coming from the SuiteSparse matrix collection [18] with various preconditioners. Our intention is to illustrate the numerical behaviors of

TABLE 4 *Set of matrices from the SuiteSparse collection used in our experiments with their associated preconditioners. The matrices are all square with real entries. n is the dimension of the matrix; Sym. is the symmetry of the matrix (1: symmetric, 0: general); Precond. is the preconditioner used;* $\kappa_2(A)$, $\kappa_2(M)$, $\kappa_2(M_L^{-1}A)$, *and* $\kappa_2(AM_R^{-1})$ *are the condition numbers of the matrices, their preconditioners, and the resulting preconditioned matrices.*

Matrix	n	Sym.	Precond.	$\kappa_2(A)$	$\kappa_2(M)$	$\kappa_2(M_L^{-1}A)$	$\kappa_2(AM_R^{-1})$
hor_131	434	0	LU - B	4.3E+04	4.3E+04	1.5E+01	1.4E+01
orsirr_1	1030	0	LU - B	7.7E+04	5.2E+05	1.1E+03	1.1E+03
bcsstk19	817	1	LU - S	1.3E+11	1.1E+11	4.7E+03	8.7E+02
bwm200	200	0	ILU - B - $t = 1e\text{-}4$	2.4E+03	2.0E+03	5.5E+01	7.6E+01
1138_bus	1138	1	ILU - S - $t = 1e\text{-}7$	8.6E+06	2.5E+05	5.9E+02	5.9E+02
young3c	841	0	ILU - S - $t = 2e\text{-}5$	9.3E+03	9.3E+03	1.5E+03	1.1E+03
gre_115	115	0	Poly. - S - $d = 20$	5.0E+01	3.4E+01	1.1E+01	1.1E+01
pores_3	532	0	Poly. - D - $d = 50$	5.6E+05	1.2E+03	8.2E+02	8.2E+02
cage5	37	0	SPAI - B - $\varepsilon = 0.5$	1.5E+01	1.1E+01	2.2E+01	1.5E+01
saylr1	238	0	SPAI - S - $\varepsilon = 0.4$	7.8E+08	5.0E+08	2.1E+04	2.1E+04

the different strategies for a variety of practical setups. We emphasize that we do not attempt a runtime comparison of the mixed precision strategies and preconditioners; this would require a dedicated study based on a careful high performance implementation which is out of the scope of this article.

We list our test set of SuiteSparse matrices and their associated preconditioners in Table 4. We use four different kinds of practical preconditioners:

- LU factors computed in low precision (i.e., bfloat16 or single precision) and implemented with the Julia standard `lu` routine. The matrices are stored in dense format before calling the `lu` routine.
- Incomplete LU factorization with dropping threshold t. We use the `IncompleteLU.jl` Julia package for the implementation.
- Arnoldi polynomial preconditioner of degree d described in [27, sect. 3.2]. We employ a homemade Julia implementation which is part of the companion Github repository[2].
- Sparse approximate inverse (SPAI) preconditioner [17, Alg. 2.1] with accuracy parameter ε. The reference [17] features two other parameters β and α that we set to $\beta = 8$ and $\alpha = n/\beta$. We use a homemade Julia implementation that can be found in the companion Github repository[2]. With this SPAI preconditioner, $M_L \neq M_R$.

In Table 4, we indicate for each matrix the preconditioner used (i.e., LU, ILU, Polynomial, or SPAI), the precision in which the preconditioner is computed (i.e., B, S, or D for bfloat16, single, or double), and the values of the parameters t, d, or ε when applicable. Note that two matrices of our set are symmetric, but are treated as unsymmetric in these experiments. In the context of our numerical analysis, these cases illustrate interesting numerical behaviors that help us compare the different mixed precision strategies. In practice, however, it is recommended to use a Krylov solver dedicated to symmetric systems over GMRES for these matrices.

In Table 5, we display the cumulated number of GMRES iterations achieved by the fourteen mixed precision variants presented in section 4 when run on the matrices listed in Table 4. For each variant we compare left-, right-, and flexible-preconditioning. The exact solution and the right-hand side are generated as described in section 4.1.2. Similarly to the random dense experiments, we use a restart

TABLE 5 *Number of cumulated GMRES iterations to reach a prescribed accuracy* $\|\widehat{x}_i - x\|_2/\|x\|_2 \leq 10^{-10}$ *for fourteen combinations of the precisions* u_g, u_a, *and* u_m *and for left-, right-, and flexible-preconditioned GMRES. For commodity, we denote in bold the results that are discussed in the text.*

	Strat.	$u_a = u_g = u_m$			$u_a = u_m \ll u_g$			$u_a = u_g \ll u_m$			$u_a \ll u_g = u_m$			$u_a \ll u_g \ll u_m$	$u_a \ll u_m \ll u_g$
		BBB	SSS	DDD	SBS	DBD	DSD	SSB	DDB	DDS	SBB	DBB	DSS	DSB	DBS
hor_131	left	**29**	13	11	**20**	19	13	**19**	20	13	**20**	19	13	18	20
	right	44	14	12	17	18	14	25	25	14	24	24	14	25	17
	flexible	47	14	12	22	22	14	16	14	12	27	27	14	16	22
orsirr_1	left	—	50	43	289	317	49	**247**	240	49	**542**	430	49	228	303
	right	—	55	38	242§	213	49	—	—	49	—	—	49	—	291§
	flexible	—	55	38	624§	523§	49	1465	260	38	—	—	49	275	883§
bcsstk19	left	—	**19**	9	**21**	13	**10**	—	—	16	—	—	**14**	—	**14**
	right	—	25	10	24§	15§	10	—	—	22	—	—	23	—	21§
	flexible	—	26	10	—	—	11	—	17103§	16	—	—	17	—	—
bwm200	left	274	39	24	**140**	**173**	38	216	183	38	268	277	38	216	174
	right	729	49	24	156	172	38	260	250	39	333	389	39	250	136
	flexible	578§	45	24	224§	203	39	**57**	29	24	239	**279**	39	**44**	191
1138_bus	left	—	34	17	825	1275	26	—	—	25	—	—	27	1412§	931
	right	—	37	18	245§	254§	25	120	125	28	—	—	31	120	328
	flexible	—	38	18	174§	169§	31	58	39	18	—	—	31	50	141§
young3c	left	—	60	33	465	407	45	—	—	60	—	—	61	—	489
	right	—	64	34	390	294	45	—	—	64	—	—	62	—	288
	flexible	—	52	34	319	368	45	1045	465	34	—	—	45	961	333
gre_115	left	461	17	13	41	35	18	138	93	17	148	148	18	138	41
	right	152	18	14	36	34	18	69	55	18	60	60	18	55	36
	flexible	43	18	14	35	38	18	19	14	14	40	40	18	20	37
pores_3	left	—	—	97	—	—	147	—	—	—	—	—	—	—	—
	right	—	—	97	—	—	148	—	—	—	—	—	—	—	—
	flexible	—	186	97	—	—	177	2510	207	133	—	—	180	1711	—
cage5	left	22	18	17	21	22	18	22	22	18	21	21	18	22	21
	right	33	22	18	37	37	22	33	33	22	37	37	22	33	37
	flexible	37	22	18	36	35	22	22	18	18	32	32	22	22	36
saylr1	left	—	215	71	—	—	174	—	—	175	—	—	181	—	—
	right	—	276	126	—	—	226	—	—	176	—	—	228	—	—
	flexible	—	274	126	—	—	227	311	140	126	—	—	227	305	—

§ Prescribed accuracy of order $\|\widehat{x} - x\|_2/\|x\|_2 \leq 10^{-10}$ is achieved but one of order 10^{-16} is impossible.

process to compute a solution with prescribed accuracy $\|\widehat{x} - x\|_2/\|x\|_2 \leq 10^{-10}$; we identically use the ten same different tolerances τ to restart GMRES and keep the run leading to the least amount of iterations. We use "—" to indicate that a given method fails to converge to the prescribed accuracy.

5.1. *Discussion of the mixed precision strategies*

We observe that the strategy $u_a = u_m \ll u_g$ (third column of Table 5) can offer substantial improvement in robustness and number of iterations. For instance, the variant BBB fails to converge for orsirr_1, bcsstk19, 1138_bus, and young3c, but the variants SBS and DBD succeed. Moreover, while BBB converges for hor_131 and bwm200, the variants SBS and DBD achieve substantially less iterations. On the other hand, we observe that SSS is (almost) always better than SBS on our test set, so that SBS achieves a tradeoff between BBB and SSS. This behavior is expected and reflects the conclusions of section 4. Comparing now DBD to SBS, we observe with bcsstk19 that DBD can exhibit a lower number of iterations than SBS, showing that DBD can be meaningful in a practical context. Yet, we also observe

cases where DBD converges more slowly than SBS; see for example bwm200 where L-DBD needs 173 iterations compare with L-SBS which needs 140 iterations. Unfortunately, we cannot interpret or anticipate these behaviors with our rounding error results. In particular, while we can often expect the convergence rate to improve when we reduce the level of rounding errors by employing variants using strictly higher precisions, it is not guaranteed and, in some instances, as we observe it here, it can even possibly increase the number of iterations. Lastly, for the matrices bcsstk19, 1138_bus, young3c, pores_3, and saylr1, we observe improvement in robustness and number of iterations when using the variant DSD instead of SSS.

As explained in section 4.2.3, flexible-preconditioned GMRES is expected to offer the best numerical behavior for the strategy $u_a = u_g \ll u_m$ (fourth column of Table 5). This is something we also conclude from our experiments with SuiteSparse matrices. Indeed, except for a few outliers, flexible-preconditioning presents better robustness and number of iterations for the variants SSB, DDB, and DDS than left- and right-preconditioning for the matrices of our set. Our results are slightly more nuanced when comparing right- and left-preconditioning than what we concluded in section 4.2.3. The matrices of our set that embody the best the conclusion of section 4.2.3 are 1138_bus and gre_115. For these matrices, flexible-preconditioning has the best number of iterations for the variants SSB and DDB, right-preconditioning is slower but converges to the prescribed accuracy, and left-preconditioning fails to converge or converges the slowest. Hence, right-preconditioned GMRES is more robust than left-preconditioned GMRES for these cases and offers a valuable alternative to flexible-preconditioning which may demand higher memory consumption. However, we can observe that this conclusion is not always true when looking at, for instance, orsirr_1 or bwm200 for which left- outperforms right-preconditioning for SSB and DDB. We emphasize that these problems do not verify $\kappa(M) \ll \kappa(A)$, which is the setup in which right-preconditioned GMRES is expected to perform better than left-preconditioned GMRES for these variants.

Consider now the strategy $u_a \ll u_g = u_m$ (fifth column of Table 5). Taking SBB first, section 4.2.4 concluded that SBB could achieve interesting tradeoffs between BBB, using strictly lower precision, and SBS or SSB, using strictly higher precision. This is also what we generally observe in our SuiteSparse experiments in Table 2. Indeed, SBB tends to be slower than SBS or SSB, but, on some problems, it converges substantially faster than BBB. For certain matrices, SBB needs approximately the same number of iterations than SBS and SSB but involves strictly lower precision. These problems are very favorable to this strategy since it means SBB should be preferred over the two other variants. We have such configuration with hor_131, where L-BBB requires 29 iterations to converge to the prescribed accuracy but L-SBB only needs 20, which is comparable to L-SBS and L-SSB which require 20 and 19, respectively. In addition to presenting a lower number of iterations than BBB, SBB can sometimes converge where BBB is unable to. This is the case for orsirr_1, where L-BBB fails to converge but L-SBB succeeds. Most of the previous comments on SBB apply equivalently to the variant DSS. In particular, the matrices bcsstk19 and 1138_bus are good examples of problems for which DSS can propose interesting tradeoffs. For instance, L-SSS needs 19 iterations for bcsstk19, L-DSD needs 10 iterations, and L-DSS falls in the middle with 14 iterations.

The remaining two strategies are the three-precision implementations. We start with $u_a \ll u_g \ll u_m$ on the penultimate column of Table 5 which features the variant DSB. For DSB to be meaningful and propose a valid tradeoff, it should converge in less iterations than variants using strictly lower precisions, such as SSB or DBB. We encounter such scenarios with matrices bwm200, 1138_bus, and pores_3. Taking bwm200 for instance, F-SSB needs 57 iterations, F-DBB needs 279 iterations, but F-DSB needs 44. Lastly, we discuss the strategy $u_a \ll u_m \ll u_g$ on the last column of Table 5, which is embodied by the variant DBS. This variant should achieve a lower number of iterations than SBS and DBB to be

considered meaningful. This is the case for bcsstk19, and to some extent 1138_bus and young3c. With bcsstk19, L-SBS needs 21 iterations, L-DBB fails to converge, but L-DBS requires 14 iterations.

5.2. *Other discussions*

In section 4, we could not observe clearly the effect of the term $u_g\kappa(AM_R^{-1})\kappa(M_R)$ in the right- and flexible-preconditioning bounds (3.23) and (3.24) with our synthetic preconditioners. With a term $u_g\kappa(M_L^{-1}A)$ instead, the left-preconditioned GMRES should exhibit better numerical behaviors on setting the precision u_g low. Interestingly, this theoretical observation may be corroborated by the problems cage5 and saylr1, both employing sparse approximate inverse preconditioners. Taking saylr1 for instance, we can see that the variants SSS, DSD, and DSS, all having $u_g = \mathrm{S}$, achieve significantly better number of iterations with left-preconditioning. The same comments apply to cage5 for the variants BBB, SBS, DBD, SBB, DBB, and DBS using $u_g = \mathrm{B}$. We can also make the observation that left-preconditioning performs better on orsirr_1 for the variants SSB, DDB, SBB, DBB, and DSB. For this matrix, however, it is unclear if it is due to the same reasons since the variants SBS and DBD also fix u_g in low precision, but right- outperforms left-preconditioning for these variants.

In section 4.2.3, where we studied the strategy $u_a = u_g \ll u_m$, we concluded from the analysis of the attainable error bounds and our experiments on synthetic problems that setting $u_g \ll u_m$ for left-preconditioned GMRES is not meaningful. However, in light of our experiments on real-life problems, we should mitigate this claim. Indeed, taking orsirr_1, L-SBB achieves 542 iterations but L-SSB reduces it to 247. We can also make a similar observation for bwm200. This behavior cannot be interpreted with our theoretical results which prescribed that setting $u_g \ll u_m$ would not improve the attainable accuracy of left-preconditioned GMRES over setting $u_g = u_m$ for u_m fixed. We often observe that the improvement or deterioration of the attainable accuracy is correlated with a reduction or an increase of the number of iterations. While it is a good empirical rule of thumb, it is not always true, as this observation seems to demonstrate.

6. Conclusion

In this study, we assessed in a comprehensive way the different mixed precision strategies that can be employed for preconditioned GMRES. To achieve this, we considered a generic three precisions layout represented by Algorithm 1, where the applications of the preconditioner to vectors are performed in precision u_m, the matrix–vector products with A are performed in precision u_a, and the rest of the GMRES operations are performed in precision u_g. We derived state-of-the-art bounds on the attainable forward errors of Algorithm 1 and validated them experimentally with experiments on synthetic problems. By analyzing these bounds and the results of our numerical experiments, we were able to identify all the numerically meaningful combinations of precisions u_g, u_m, and u_a; there are listed in Table 2 (and in Table 3). Several of these strategies are new and present combinations of precisions that are numerically meaningful and have never appeared in the literature. Moreover, we made substantial contributions to the other already existing strategies. First, each of the already existing strategies was dedicated to one preconditioning technique: either left-, right-, or flexible-preconditioning. We broadened their scope to the three preconditioning techniques considered in this article. Second, the existing strategy $u_a = u_m \ll u_g$ had previously only been studied with specific preconditioners, such as LU or sparse approximate inverse preconditioners. We broadened the theory and the use of this strategy to any preconditioner. We finally validated the different mixed precision strategies on a set of real-life problems from the SuiteSparse collection using different practical preconditioners.

Importantly, we uncovered critical differences in robustness and accuracy between left-, right-, and flexible-preconditioning for a same given set of precisions u_g, u_m, and u_a. We derived indications based on the analysis of our error bounds that prescribe which preconditioning technique may be best for a given problem and set of precisions. These indications appeared reliable when challenged against our experiments on synthetic and real-life problems. It should be noted that we observed a few outliers. Those tend to be cases that cannot be described by our simplified bounds (3.22) to (3.24), and for which we need to fall back on the original bounds (3.13), (3.17), and (3.21) to look for theoretical insights. It may also be cases offering a poor correlation between the level of rounding error, which we monitor with our theory, and the convergence rate of GMRES.

In light of the results derived in this article, we finish by outlining some open issues that might be the topic of future work:

- For conciseness, we focus this article on the study of the forward error. We expect most of our important conclusions regarding the numerical behaviors of the different mixed precision strategies to extend relatively straightforwardly in backward error. However, we recognize that a dedicated analysis of the backward error is of interest and may uncover more specific results.
- We confined the scope of this study to a numerical assessment of mixed precision strategies for preconditioned GMRES. This is a first necessary step towards implementing some of these mixed precision approaches in industrial software, which will require a separate study focusing on the performance analysis of these strategies to fully validate their effectiveness and practicality.

Funding

The work of the first author has benefited from a national grant managed by the French National Research Agency (Agence Nationale de la Recherche) attributed to the Exa-SofT project of the NumPEx PEPR program, under the reference ANR-22-EXNU-0003.

The work of the second author was supported in part by the National Natural Science Foundation of China (Grants 12125108, 12021001, 12288201) and the RGC grant JLFS/P-501/24 for the CAS AMSS-PolyU Joint Laboratory in Applied Mathematics.

The work of the third author was supported by the InterFLOP (ANR-20-CE46-0009), MixHPC (ANR-23-CE46-0005-01), and NumPEx ExaMA (ANR-22-EXNU-0002) projects of the French National Agency for Research (ANR).

The work of the fourth author was supported by the National Natural Science Foundation of China (No. 12288201 and W2433016).

A. Proof of Theorem 3.1

We start the proof by taking into account the specific features of the MGS orthogonalization and wish to identify a dimension $k \leq n$ of the basis $\widetilde{Z}_k$ for which we can prove that the computed solution has achieved a small forward error satisfying (3.8). We recall the result of the reasoning [11, eq. (5.15) to (5.17)], which was derived for GMRES with MGS orthogonalization, and that showed that for any nonsingular basis $\widetilde{Z}_k$ of increasing dimension, there exists a dimension $k \leq n$ at which we satisfy for all $\phi > 0$

$$\text{(if } k \leq n-1) \quad \sigma_{\min}\big(\big[M_L^{-1}b\phi,\, M_L^{-1}A\widetilde{Z}_k\big]\big) < c(n,k)\big(u_g+\varepsilon_c\big)\big\|\big[M_L^{-1}b\phi,\, M_L^{-1}A\widetilde{Z}_k\big]\big\|_F, \tag{A.1}$$

and

$$\sigma_{\min}^{-1}(\widehat{V}_k) \leq 4/3 \quad \text{and} \quad \sigma_{\max}(\widehat{V}_k) \leq 4/3,$$

where $\widehat{V}_{k+1} = [\widehat{V}_k, \widehat{v}_{k+1}]$ is obtained by the MGS orthogonalization of $[\widehat{b}, \widehat{C}_k]$, and where $\widehat{b} = \mathrm{fl}(M_L^{-1}b)$ and $\widehat{C}_k = \mathrm{fl}(M_L^{-1}A\widetilde{Z}_k)$ are defined in (3.2) and (3.3). Actually, the reasoning [11, eq. (5.15) to (5.17)] holds for $\widetilde{Z}_k \equiv \widehat{V}_k$ and $M_L \equiv I$, but can be straightforwardly adapted to $\widetilde{Z}_k \neq \widehat{V}_k$ and $M_L \neq I$ which we consider here. We take this dimension k to be the one of Theorem 3.1 and, therefore, (3.7) is satisfied. The remainder of the proof will be about to show that Algorithm 2 computes a solution $\widehat{x}_k$ satisfying (3.8) for this dimension k and basis $\widetilde{Z}_k$ under (A.1) and the assumptions of Theorem 3.1.

The rest of the proof consists in rewriting the reasoning derived to prove [11, Thm. 3.1]. Specifically, the relevant changes allowing for the betterment of the bound will mostly concern the third and fourth parts of this former proof. The two first parts are only affected by the use of the MGS orthogonalization, where [11] uses less stringent assumptions on how Algorithm 2 is implemented. In particular, [11] uses an accuracy parameter noted $\varepsilon_{\mathrm{ls}}$ to keep a certain level of abstraction on how the least squares problem at line 3 of Algorithm 2 is solved. Because we use the classic MGS approach in this article, we know that the accuracy of this operation satisfies $\varepsilon_{\mathrm{ls}} \equiv c(n,k)u_g$; see [11, sect. 5.3]. For this reason, adapting the reasoning of these two first parts is trivial and holds under conditions (3.2) to (3.6) and (A.1), and we do not present the details. We adopt the same notation as in [11] when possible for the ease of comparison. We recall from [11, eqs. (3.13) and (3.14)] that the least squares residual $\bar{r}_k$ associated to the computed least squares solution $\widehat{y}_k$ at line 3 of Algorithm 2 satisfies

$$\begin{aligned} \bar{r}_k &= M_L^{-1}b + \Delta\widetilde{b}^{(1)} - (M_L^{-1}A\widetilde{Z}_k + \Delta C_k)\widehat{y}_k, \\ \|\Delta C_k\|_F \lesssim c(n,k)(\varepsilon_c + u_g)\|M_L^{-1}A\widetilde{Z}_k\|_F, &\qquad \|\Delta\widetilde{b}^{(1)}\|_2 \lesssim c(n,k)(\varepsilon_b + u_g)\|M_L^{-1}b\|_2. \end{aligned} \tag{A.2}$$

From [11, eq. (3.23)], the least squares residual can be bounded in norm as

$$\|\bar{r}_k\|_2 \lesssim c(n,k)(\varepsilon_c + \varepsilon_b + u_g)\|M_L^{-1}A\widetilde{Z}_k\|_F\|\widehat{y}_k\|_2. \tag{A.3}$$

In order to sharpen the forward error bounds of [11, Thm. 3.1] when a right-preconditioner is used, we shall not bound the errors of the left-preconditioned linear system directly as done previously, but rather the one of the split-preconditioned linear system $M_L^{-1}AM_R^{-1}u = M_L^{-1}b$. To do so, we introduce the following quantity

$$\hat{x}_k^R = M_R\widetilde{Z}_k\widehat{y}_k, \tag{A.4}$$

where $M_R \in \mathbb{R}^{n\times n}$ is nonsingular. Note that we use a "hat" in the notation $\hat{x}_k^R$ to signify that this quantity is linked to the computed $\widehat{y}_k$ compared with x^R which is the exact solution of the split-preconditioned system. Nevertheless, we emphasize that $\hat{x}_k^R$ is not technically a computed quantity but a mathematical artifice that does not exist at the algorithm level. We define

$$\Delta\widetilde{A}^{(1)} = \Delta C_k\widehat{y}_k\|\hat{x}_k^R\|_2^{-2}(\hat{x}_k^R)^T$$

which gives using (A.4)

$$(M_L^{-1}AM_R^{-1} + \Delta\widetilde{A}^{(1)})\hat{x}_k^R = M_L^{-1}AM_R^{-1}\hat{x}_k^R + \Delta\widetilde{A}^{(1)}\hat{x}_k^R = (M_L^{-1}A\widetilde{Z}_k + \Delta C_k)\widehat{y}_k.$$

The residual defined in (A.2) can then be expressed as

$$\bar{r}_k = M_L^{-1}b + \Delta\widetilde{b}^{(1)} - (M_L^{-1}AM_R^{-1} + \Delta\widetilde{A}^{(1)})\hat{x}_k^R, \tag{A.5}$$

where

$$\|\Delta\widetilde{A}^{(1)}\|_F \lesssim c(n,k)(\varepsilon_c + u_g)\|M_L^{-1}A\widetilde{Z}_k\|_F\|\widehat{y}_k\|_2/\|\hat{x}_k^R\|_2. \tag{A.6}$$

We now form the quantities

$$\Delta\widetilde{b}^{(2)} = -\frac{\|M_L^{-1}b\|_2}{\|M_L^{-1}AM_R^{-1}\|_F\|\hat{x}_k^R\|_2 + \|M_L^{-1}b\|_2}\bar{r}_k$$

and

$$\Delta\widetilde{A}^{(2)} = \frac{\|M_L^{-1}AM_R^{-1}\|_F\|\hat{x}_k^R\|_2}{\|M_L^{-1}AM_R^{-1}\|_F\|\hat{x}_k^R\|_2 + \|M_L^{-1}b\|_2}\bar{r}_k\frac{(\hat{x}_k^R)^T}{\|\hat{x}_k^R\|_2^2}$$

satisfying $\bar{r}_k = \Delta\widetilde{A}^{(2)}\hat{x}_k^R - \Delta\widetilde{b}^{(2)}$ and which can be bounded using (A.3) such that

$$\begin{aligned}\|\Delta\widetilde{b}^{(2)}\|_2 &\lesssim c(n,k)(\varepsilon_c+\varepsilon_b+u_g)\alpha'\|M_L^{-1}b\|_2,\\ \|\Delta\widetilde{A}^{(2)}\|_F &\lesssim c(n,k)(\varepsilon_c+\varepsilon_b+u_g)\alpha'\|M_L^{-1}AM_R^{-1}\|_F,\end{aligned} \tag{A.7}$$

with

$$\alpha' = \frac{\|M_L^{-1}A\widetilde{Z}_k\|_F}{\|M_L^{-1}AM_R^{-1}\|_F}\frac{\|\widehat{y}_k\|_2}{\|\hat{x}_k^R\|_2}.$$

Finally, by replacing $\bar{r}_k$ by $\Delta\widetilde{A}^{(2)}\hat{x}_k^R - \Delta\widetilde{b}^{(2)}$ in (A.5), we can conclude that $\hat{x}_k^R$ is the exact solution of the perturbed linear system $(M_L^{-1}AM_R^{-1} + \Delta\widetilde{A})\hat{x}_k^R = M_L^{-1}b + \Delta\widetilde{b}$ where

$$\Delta\widetilde{A} \equiv \Delta\widetilde{A}^{(1)} + \Delta\widetilde{A}^{(2)} \qquad \text{and} \qquad \Delta\widetilde{b} \equiv \Delta\widetilde{b}^{(1)} + \Delta\widetilde{b}^{(2)}.$$

In addition, from the error bounds (A.2), (A.6), and (A.7), the errors $\Delta\widetilde{A}$ and $\Delta\widetilde{b}$ satisfy

$$\begin{aligned}\|\Delta\widetilde{A}\|_F &\lesssim c(n,k)(\alpha'\varepsilon_c+\alpha'\varepsilon_b+\alpha'u_g)\|M_L^{-1}AM_R^{-1}\|_F,\\ \|\Delta\widetilde{b}\|_F &\lesssim c(n,k)(\alpha'\varepsilon_c+\beta'\varepsilon_b+\beta'u_g)\|M_L^{-1}b\|_2,\end{aligned}$$

with

$$\beta' = \max\left(1,\alpha'\right).$$

The backward error of the split-preconditioned system therefore satisfies the bound

$$\frac{\|M_L^{-1}b - M_L^{-1}AM_R^{-1}\hat{x}_k^R\|_2}{\|M_L^{-1}AM_R^{-1}\|_F\|\hat{x}_k^R\|_2 + \|M_L^{-1}b\|_2} \lesssim c(n,k)\xi', \tag{A.8}$$

with

$$\xi' = \alpha'\varepsilon_c + \beta'\varepsilon_b + \beta'u_g.$$

To obtain an upper bound of $\|\widehat{y}_k\|_2$ which is function of $\|\hat{x}_k^R\|_2$, we use the relation $\hat{x}_k^R = M_R\widetilde{Z}_k\widehat{y}_k$ from which we can conclude that

$$\|\hat{x}_k^R\|_2 = \|M_R\widetilde{Z}_k\widehat{y}_k\|_2 \geq \min_y\frac{\|M_R\widetilde{Z}_k y\|_2}{\|y\|_2}\|\widehat{y}_k\|_2 \geq \sigma_{\min}(M_R\widetilde{Z}_k)\|\widehat{y}_k\|_2,$$

and which allows us to rework (A.8) as

$$\frac{\|M_L^{-1}b - M_L^{-1}AM_R^{-1}\hat{x}_k^R\|_2}{\|M_L^{-1}AM_R^{-1}\|_F\|\hat{x}_k^R\|_2 + \|M_L^{-1}b\|_2} \lesssim c(n,k)\xi'', \qquad \xi'' = \alpha''\varepsilon_c + \beta''\varepsilon_b + \beta''u_g, \tag{A.9}$$

where

$$\alpha'' = \sigma_{\min}^{-1}(M_R\widetilde{Z}_k)\frac{\|M_L^{-1}A\widetilde{Z}_k\|_F}{\|M_L^{-1}AM_R^{-1}\|_F}, \qquad \beta'' = \max\left(1,\alpha''\right).$$

Note that $M_R\widetilde{Z}_k$ is full-rank and the expressions above are well-defined since M_R is nonsingular and $\widetilde{Z}_k$ is full-rank by assumption.

The last part of the proof consists of deriving bounds on the forward and backward errors of the original system. We can bound the forward error of the split-preconditioned system with the backward error (A.9) and the condition number of $M_L^{-1}AM_R^{-1}$. We obtain

$$\frac{\|\hat{x}_k^R - x^R\|_2}{\|x^R\|_2} \lesssim c(n,k)\xi''\kappa\big(M_L^{-1}AM_R^{-1}\big). \tag{A.10}$$

Because the exact solution x^R of the split preconditioned system is $x^R = M_R x$ where x is the exact solution of the original system $Ax = b$, we can derive the forward error of $Ax = b$ from (A.10). Using (3.4), we can connect $\hat{x}_k^R$ and $\widehat{x}_k$ by

$$\widehat{x}_k = \widetilde{Z}_k\widehat{y}_k + \Delta_x = M_R^{-1}\hat{x}_k^R + \Delta_x,$$

which gives, using (A.10),

$$\begin{aligned}\|\widehat{x}_k - x\|_2 &= \|M_R^{-1}\hat{x}_k^R + \Delta_x - M_R^{-1}x^R\| \le \|M_R^{-1}\|_F\|\hat{x}_k^R - x^R\|_2 + \|\Delta_x\|_2\\ &\lesssim c(n,k)\xi''\kappa\big(M_L^{-1}AM_R^{-1}\big)\|M_R^{-1}\|_F\|x^R\|_2 + \varepsilon_x\|\widetilde{Z}_k\widehat{y}_k\|_2.\end{aligned}$$

Remarking from (3.4) that

$$\|\widetilde{Z}_k\widehat{y}_k\|_2 \le \big(\|\widehat{x}_k\|_2 + \|\Delta_x\|_2\big) \le \big(\|\widehat{x}_k\|_2 + \varepsilon_x\|\widetilde{Z}_k\widehat{y}_k\|_2\big),$$

giving

$$\|\widetilde{Z}_k\widehat{y}_k\|_2 \le \big(1-\varepsilon_x\big)^{-1}\|\widehat{x}_k\|_2 \lesssim \big(\|\widehat{x}_k - x\|_2 + \|x\|_2\big),$$

we finally obtain by dropping second order terms and using $\|x^R\|_2 \le \|M_R\|_F\|x\|_2$

$$\frac{\|\widehat{x}_k - x\|_2}{\|x\|_2} \lesssim c(n,k)\xi''\kappa\big(M_L^{-1}AM_R^{-1}\big)\kappa\big(M_R\big) + \varepsilon_x, \tag{A.11}$$

and we recover (3.8).

B. Proof of Theorem 3.2

To prove Theorem 3.2, we use Theorem 3.1, which requires us to identify the accuracy parameters ε_c, ε_b, and ε_x in the error models (3.2) to (3.4), and to ensure that the conditions (3.5) and (3.6) are met at an iteration $k \le n$ satisfying (3.7).

We start by identifying ε_c in the error model (3.2). From (3.9) and [23, eq. (3.11)], the computed matrix–vector products $(M_L^{-1}A)\widehat{v}_j$ satisfy for all $j \leq k$

$$\begin{aligned} \mathrm{fl}\left(M_L^{-1}A\widehat{v}_j\right) &= \left(M_L^{-1} + \Delta M_L^{(j)}\right)\left(A + \Delta A^{(j)}\right)\widehat{v}_j, \\ |\Delta A^{(j)}| &\lesssim c(n,k)u_a|A|. \end{aligned}$$

Using the above and dropping the second-order term $\Delta M_L^{(j)}\Delta A^{(j)}\widehat{v}_j$, we could characterize the error Δ_c in the rounding error model (3.2) generated by the kernel with

$$\begin{aligned} \mathrm{fl}\left(M_L^{-1}A\widehat{v}_j\right) \approx M_L^{-1}A\widehat{v}_j + \Delta_c^{(j)}, \qquad \Delta_c^{(j)} = M_L^{-1}\Delta A^{(j)}\widehat{v}_j + \Delta M_L^{(j)}A\widehat{v}_j, \\ \Delta_c = [\Delta_c^{(1)}, \ldots, \Delta_c^{(k)}]. \end{aligned}$$

However, this will not lead to the sharpest forward error bound. Instead, we will use a trick that consists in inserting the rounding errors inside the search basis $\widetilde{Z}_k$ by writing

$$\mathrm{fl}\left(M_L^{-1}A\widehat{v}_j\right) = M_L^{-1}A\left(I + A^{-1}\Delta A^{(j)} + A^{-1}M_L\Delta M_L^{(j)}A + A^{-1}M_L\Delta M_L^{(j)}\Delta A^{(j)}\right)\widehat{v}_j$$

from which we define

$$\begin{aligned} \widetilde{z}_j &= \left(I + A^{-1}\Delta A^{(j)} + A^{-1}M_L\Delta M_L^{(j)}A + A^{-1}M_L\Delta M_L^{(j)}\Delta A^{(j)}\right)\widehat{v}_j \\ \widetilde{Z}_k &= \left[\widetilde{z}_1, \ldots, \widetilde{z}_k\right], \qquad \mathrm{fl}\left(M_L^{-1}A\widehat{v}_j\right) = M_L^{-1}A\widetilde{z}_j. \end{aligned} \tag{B.1}$$

Defining the basis $\widetilde{Z}_k$ in such a way yields $\Delta_c \equiv 0$ in the rounding error model (3.2) and, so, $\varepsilon_c \equiv 0$. In some sense, we pick $\widetilde{Z}_k$ such that the product $\mathrm{fl}(M_L^{-1}A\widetilde{Z}_k)$ is computed exactly. However, using this trick does not withdraw the errors generated by the application of A and M_L^{-1}, but rather delays their effect to the computation of the solution approximation at line 4 of Algorithm 2. For readability, we will consider in the rest of the text that $A^{-1}M_L\Delta M_L^{(j)}\Delta A^{(j)}$ is a second order term in (B.1); this is true under assumption (3.12).

The computation of the preconditioned right-hand side corresponds to one application of the preconditioner M_L to the vector b yielding an error we note $\Delta M_L^{(b)}$ and which satisfies (3.9). Hence, we characterize the error Δ_b in the error model (3.3) with

$$\mathrm{fl}\left(M_L^{-1}b\right) = M_L^{-1}b + \Delta_b, \qquad \Delta_b \equiv \Delta M_L^{(b)}b, \qquad \|\Delta_b\|_2 \leq u_m\rho_b^L\|M_L^{-1}b\|_2, \tag{B.2}$$

where $u_m\rho_b^L$ is defined by (3.11). From (B.2), we identify $\varepsilon_b \equiv u_m\rho_b^L$.

We finally identify ε_x in the model (3.4). Because $\widehat{x}_k$ is computed through a standard matrix–vector product with $\widehat{V}_k$ in precision u_g satisfying [23, eq. (3.11)], it yields

$$\widehat{x}_k = \mathrm{fl}(\widehat{V}_k\widehat{y}_k) = (\widehat{V}_k + \Delta V_k)\widehat{y}_k, \qquad \|\Delta V_k\|_F \lesssim c(n,k)u_g. \tag{B.3}$$

In the above, we used the fact that the vectors $\widehat{v}_j$ are normalized in precision u_g such that

$$\|\widehat{v}_j\|_2 \approx 1 \quad \text{and} \quad \|\widehat{V}_k\|_F \approx k^{1/2}. \tag{B.4}$$

As the model (3.4) is expressed with the basis $\widetilde{Z}_k \neq \widehat{V}_k$, we need to relate $\widetilde{Z}_k$ to $\widehat{V}_k$. Using (B.1) and (B.4), we have

$$\widetilde{z}_j = \widehat{v}_j + \Delta z_j, \qquad \|\Delta z_j\|_2 \lesssim c(n,k)u_a\kappa\big(A\big) + u_m\rho_M^L,$$

where $u_m\rho_M^L$ is defined by (3.10). We do not further simplify the term $u_m\rho_M^L$ using the sub-multiplicative norm inequality since, in some instances, the form of the error $\Delta M_L^{(j)}$ can lead to further simplification and the bound would be too crude. Noting $\Delta Z_k = [\Delta z_1, \ldots, \Delta z_k]$, we can finally write

$$\widetilde{Z}_k = \widehat{V}_k + \Delta Z_k, \qquad \|\Delta Z_k\|_F \lesssim c(n,k)\left(u_a\kappa\big(A\big) + u_m\rho_M^L\right). \tag{B.5}$$

Hence, going back to (B.3) and using (B.5), we have

$$\begin{aligned} \widehat{x}_k &= \widetilde{Z}_k\widehat{y}_k + \Delta_x, \qquad \Delta_x = (\Delta V_k - \Delta Z_k)\,\widehat{y}_k \\ \|\Delta_x\|_2 &\lesssim c(n,k)(u_g + u_a\kappa\big(A\big) + u_m\rho_M^L)\|\widehat{y}_k\|_2, \end{aligned} \tag{B.6}$$

from which we identify

$$\varepsilon_x \equiv c(n,k)(u_g + u_a\kappa\big(A\big) + u_m\rho_M^L)\frac{\|\widehat{y}_k\|_2}{\|\widetilde{Z}_k\widehat{y}_k\|_2}.$$

The expression above can be slightly simplified by using (3.7) and (B.5) to observe that

$$\sigma_{\min}(\widetilde{Z}_k) \geq \sigma_{\min}(\widehat{V}_k) - \|\Delta Z_k\|_F \geq 3/4 - c(n,k)(u_a\kappa\big(A\big) + u_m\rho_M^L), \tag{B.7}$$

which, from assumption (3.12), gives

$$\frac{\|\widehat{y}_k\|_2}{\|\widetilde{Z}_k\widehat{y}_k\|_2} \leq \frac{\|\widehat{y}_k\|_2}{\sigma_{\min}(\widetilde{Z}_k)\|\widehat{y}_k\|_2} \lesssim 4/3,$$

so that $\varepsilon_x \lesssim c(n,k)(u_g + u_a\kappa\big(A\big) + u_m\rho_M^L)$. Under assumption (3.12), we also have $\varepsilon_x \ll 1$ and we guarantee condition (3.6).

We now wish to show that condition (3.5) is met for the previously identified ε_c and ε_b. Remarking that $\sigma_{\min}(M_L^{-1}A\widetilde{Z}_k) \geq \sigma_{\min}(M_L^{-1}A)\sigma_{\min}(\widetilde{Z}_k)$, using (B.7), assumption (3.12) to drop second order terms, and the fact that $\|\widetilde{Z}_k\|_F \approx \|\widehat{V}_k\|_F \approx k^{1/2}$ from (B.4) and (B.5), we obtain

$$\sigma_{\min}(M_L^{-1}A\widetilde{Z}_k) \geq \sigma_{\min}(M_L^{-1}A)\sigma_{\min}(\widetilde{Z}_k) \gtrsim 3\sigma_{\min}(M_L^{-1}A)/4$$

and

$$(\varepsilon_c + \varepsilon_b + u_g)\|M_L^{-1}A\widetilde{Z}_k\|_F \lesssim c(n,k)\left(u_g + u_m\rho_b^L\right)\|M_L^{-1}A\|_F,$$

and condition (3.5) is met under assumption (3.12).

Hence, Theorem 3.1 guarantees that there exists an iteration $k \leq n$ at which the forward error bound (3.8) holds. To recover the bound (3.13), we can simplify the expression α, β, and λ in (3.8) by replacing $M_R \equiv I$, using $\|\widetilde{Z}_k\|_F \approx k^{1/2}$, and using (B.7). It then comes that

$$\begin{aligned} \alpha\varepsilon_c = 0, \qquad \beta\varepsilon_b \lesssim c(n,k)u_m\rho_b^L, \qquad \beta u_g \lesssim c(n,k)u_g, \\ \lambda\varepsilon_x \lesssim c(n,k)\left(u_g + u_a\kappa\big(A\big) + u_m\rho_M^L\right)/\kappa\big(M_L^{-1}A\big), \end{aligned}$$

and we recover (3.13).

C. Proof of Theorem 3.3

The proof follows the same structure as the one of Theorem 3.2. Hence, we wish to identify ε_c, ε_b, and ε_x, and ensure that the conditions (3.5) and (3.6) are met at any iteration $k \leq n$ satisfying (3.7) to invoke the forward error bound (3.8).

We first study the matrix–matrix product $A(M_R^{-1}\widehat{V}_k)$ and identify ε_c in the model (3.2). Noting $\widehat{z}_j = \mathrm{fl}(M_R^{-1}\widehat{v}_j)$ and $\widehat{Z}_k = [\widehat{z}_1, \ldots, \widehat{z}_k]$, and using [23, eq. (3.11)], the right-preconditioned matrix–vector product kernel satisfies

$$\mathrm{fl}\left(AM_R^{-1}\widehat{v}_j\right) = \left(A + \Delta A^{(j)}\right)\widehat{z}_j, \qquad |\Delta A^{(j)}| \lesssim c(n,k)u_a|A|. \tag{C.1}$$

Similarly to the left-preconditioned GMRES analysis (see Appendix B), we insert the rounding errors generated by the matrix–vector product kernel inside the search space $\widetilde{Z}_k$. Doing so is fundamental to obtaining the sharpest and most descriptive forward error bound. We have

$$\widetilde{z}_j = \widehat{z}_j + A^{-1}\Delta A^{(j)}\widehat{z}_j, \qquad \widetilde{Z}_k = \left[\widetilde{z}_1, \ldots, \widetilde{z}_k\right], \qquad \mathrm{fl}(AM_R^{-1}\widehat{v}_j) = A\widetilde{z}_j. \tag{C.2}$$

The difference between $\widehat{Z}_k$ and $\widetilde{Z}_k$ is that the vectors of $\widehat{Z}_k$ are the vectors computed by Algorithm 1 (right) at line 6, and are formed and stored at some point in memory. On the other hand, the vectors of $\widetilde{Z}_k$ are an artifact of our mathematical reasoning and are not accessible algorithmically. With such a $\widetilde{Z}_k$, we have $\Delta_c \equiv 0$ in the rounding error model (3.2) and $\varepsilon_c \equiv 0$.

Now, consider the computation of the solution approximation $\widehat{x}_k$, which is computed through a standard matrix–vector product with $\widehat{V}_k$ in precision u_g followed by the application of the right-preconditioner M_R in precision u_m yielding an error $\Delta M_R^{(x)}$ satisfying (3.14). Dropping second order terms and using (3.14) and [23, eq. (3.11)] gives

$$\begin{aligned}\widehat{x}_k &= \mathrm{fl}\left(M_R^{-1}\widehat{V}_k\widehat{y}_k\right) = (M_R^{-1} + \Delta M_R^{(x)})(\widehat{V}_k + \Delta V_k)\widehat{y}_k \\ &\approx \left(M_R^{-1}\widehat{V}_k + \Delta M_R^{(x)}\widehat{V}_k + M_R^{-1}\Delta V_k\right)\widehat{y}_k,\end{aligned} \tag{C.3}$$

where $\|\Delta V_k\|_F \lesssim c(n,k)u_g\|\widehat{V}_k\|_F$. To relate the above to the model (3.4), we need to relate $\widehat{Z}_k$ and $\widetilde{Z}_k$. Using (3.14) and (B.4), we get

$$\widehat{Z}_k = \mathrm{fl}(M_R^{-1}\widehat{V}_k) = M_R^{-1}\widehat{V}_k + \Delta Z_k^{(1)}, \quad \|\Delta Z_k^{(1)}\|_F \lesssim c(n,k)u_m\eta_R\|M_R^{-1}\|_F. \tag{C.4}$$

In addition, using (B.4) and assumption (3.16) which guarantees in particular $u_m\eta_R \ll 1$, we also obtain

$$\|\widehat{Z}_k\|_F \leq \|M_R^{-1}\|_F\|\widehat{V}_k\|_F + \|\Delta Z_k^{(1)}\|_F \lesssim k^{\frac{1}{2}}\|M_R^{-1}\|_F\left(1 + c(n,k)u_m\eta_R\right) \approx k^{\frac{1}{2}}\|M_R^{-1}\|_F.$$

Using (C.1) and (C.2), we can link $\widetilde{Z}_k$ and $\widehat{Z}_k$, we have

$$\begin{aligned}\widetilde{Z}_k = \widehat{Z}_k + \Delta Z_k^{(2)}, \qquad \Delta Z_k^{(2)} &= [A^{-1}\Delta A^{(1)}\widehat{z}_1, \ldots, A^{-1}\Delta A^{(k)}\widehat{z}_k], \\ |\Delta Z_k^{(2)}| &\lesssim c(n,k)u_a|A^{-1}||A||\widehat{Z}_k|, \\ \|A\Delta Z_k^{(2)}\|_F &\lesssim c(n,k)u_a\|A\|_F\|\widehat{Z}_k\|_F \lesssim c(n,k)u_a\|A\|_F\|M_R^{-1}\|_F, \\ \|M_R\Delta Z_k^{(2)}\|_F &\lesssim c(n,k)u_a\|M_RA^{-1}\|_F\|A\|_F\|M_R^{-1}\|_F.\end{aligned} \tag{C.5}$$

Hence, it follows from (C.3) and (C.4) that

$$\widehat{x}_k \approx \widetilde{Z}_k \widehat{y}_k + \Delta_x, \qquad \Delta_x = \left(\Delta M_R^{(x)} \widehat{V}_k + M_R^{-1} \Delta V_k - \Delta Z_k^{(1)} - \Delta Z_k^{(2)} \right) \widehat{y}_k. \tag{C.6}$$

To bound $\|\Delta_x\|_2$ we need further developments. From (C.5), we write

$$\|\Delta Z_k^{(2)} \widehat{y}_k\|_2 \leq c(n,k) u_a \kappa(A) \rho_A^R \|\widehat{Z}_k \widehat{y}_k\|_2, \tag{C.7}$$

where ρ_A^R is defined in (3.15). Under assumption (3.16) guaranteeing $u_a \kappa(A) \rho_A^R \ll 1$, by using the relation (C.5) and writing

$$\|\widehat{Z}_k \widehat{y}_k\|_2 - \|\Delta Z_k^{(2)} \widehat{y}_k\|_2 \leq \|\widetilde{Z}_k \widehat{y}_k\|_2 \leq \|\widehat{Z}_k \widehat{y}_k\|_2 + \|\Delta Z_k^{(2)} \widehat{y}_k\|_2, \tag{C.8}$$

we deduce $\|\widehat{Z}_k \widehat{y}_k\|_2 \approx \|\widetilde{Z}_k \widehat{y}_k\|_2$. Using (C.4) and assumption (3.16), which guarantees $u_m \eta_R \kappa(M_R) \ll 1$ leading to $\sigma_{\min}(M_R^{-1}) \gg u_m \eta_R \|M_R^{-1}\|_F$, gives

$$\sigma_{\min}(\widehat{Z}_k) \geq \sigma_{\min}(M_R^{-1}) \sigma_{\min}(\widehat{V}_k) - \|\Delta Z_k^{(1)}\|_F \gtrsim 3\sigma_{\min}(M_R^{-1})/4. \tag{C.9}$$

Hence, from (C.9) and using $\|\widehat{Z}_k \widehat{y}_k\|_2 \approx \|\widetilde{Z}_k \widehat{y}_k\|_2$, we observe that

$$\frac{\|M_R^{-1}\|_F \|\widehat{y}_k\|_2}{\|\widetilde{Z}_k \widehat{y}_k\|_2} \approx \frac{\|M_R^{-1}\|_F \|\widehat{y}_k\|_2}{\|\widehat{Z}_k \widehat{y}_k\|_2} \leq \frac{\|M_R^{-1}\|_F \|\widehat{y}_k\|_2}{\sigma_{\min}(\widehat{Z}_k) \|\widehat{y}_k\|_2} \lesssim c(n,k) \kappa(M_R), \tag{C.10}$$

from which, using (3.14), (B.4), (C.3), and (C.4), we deduce

$$\| \left(\Delta M_R^{(x)} \widehat{V}_k + M_R^{-1} \Delta V_k - \Delta Z_k^{(1)} \right) \widehat{y}_k \|_2 \lesssim c(n,k) \left(u_m \eta_R + u_g \right) \kappa(M_R) \|\widetilde{Z}_k \widehat{y}_k\|_2.$$

Finally, using the bound (C.7) together with $\|\widehat{Z}_k \widehat{y}_k\|_2 \approx \|\widetilde{Z}_k \widehat{y}_k\|_2$ in (C.6), we obtain

$$\|\Delta_x\|_2 \lesssim c(n,k) \left(u_m \eta_R \kappa(M_R) + u_g \kappa(M_R) + u_a \kappa(A) \rho_A^R \right) \|\widetilde{Z}_k \widehat{y}_k\|_2, \tag{C.11}$$

and we identify

$$\varepsilon_x \equiv c(n,k) \left(u_m \eta_R \kappa(M_R) + u_g \kappa(M_R) + u_a \kappa(A) \rho_A^R \right)$$

in the model (3.4). Moreover, under assumption (3.16), we have $\varepsilon_x \ll 1$ and we meet condition (3.6).

As there is no left-preconditioner, there is no error in forming the left-preconditioned right-hand side at line 2 of Algorithm 2; we have $\varepsilon_b \equiv 0$.

Because $\varepsilon_c = \varepsilon_b = 0$, showing that condition (3.5) is met reduces to show that $u_g \kappa(A\widetilde{Z}_k) \ll 1$. Combining (C.4) and (C.5), we have

$$\widetilde{Z}_k = M_R^{-1}\widehat{V}_k + \Delta Z_k^{(1)} + \Delta Z_k^{(2)} \tag{C.12}$$

leading to

$$\begin{aligned}\sigma_{\min}(A\widetilde{Z}_k) &\geq \sigma_{\min}(AM_R^{-1})\sigma_{\min}(\widehat{V}_k + M_R\Delta Z_k^{(1)} + M_R\Delta Z_k^{(2)}) \\ &\geq \sigma_{\min}(AM_R^{-1})\left(\sigma_{\min}(\widehat{V}_k) - \|M_R\Delta Z_k^{(1)}\|_F - \|M_R\Delta Z_k^{(2)}\|_F\right).\end{aligned}$$

Using (3.7), remarking that

$$\frac{\|A\|_F\|M_R^{-1}\|_F}{\|AM_R^{-1}\|_F} \leq \min\left(\kappa_F(A), \kappa_F(M_R)\right),$$

using (C.4) and (C.5) to bound respectively $\|M_R\Delta Z_k^{(1)}\|_F$ and $\|M_R\Delta Z_k^{(2)}\|_F$, and using assumption (3.16) to drop second order terms, we obtain

$$\begin{aligned}\sigma_{\min}(A\widetilde{Z}_k) &\gtrsim \sigma_{\min}(AM_R^{-1})\left(3/4 - c(n,k)\left[u_m\eta_R\kappa(M_R) + u_a\kappa(AM_R^{-1})\kappa(M_R)\right]\right) \\ &\approx 3\sigma_{\min}(AM_R^{-1})/4.\end{aligned} \tag{C.13}$$

On the other hand, using (B.4) , (C.4), (C.5), and (C.12), we bound

$$\begin{aligned}\|A\widetilde{Z}_k\|_F &\leq \|AM_R^{-1}\|\|\widehat{V}_k\|_F + \|A\Delta Z_k^{(1)}\|_F + \|A\Delta Z_k^{(2)}\| \\ &\lesssim c(n,k)\left(1 + u_m\eta_R\kappa(M_R) + u_a\kappa(A)\right)\|AM_R^{-1}\|_F.\end{aligned} \tag{C.14}$$

Dropping second order terms with assumption (3.16) , we get $\|A\widetilde{Z}_k\|_F \lesssim c(n,k)\|AM_R^{-1}\|_F$ which, combined together with (C.13), assures that condition (3.5) is met under assumption (3.16).

We showed that the conditions of Theorem 3.1 are met for any iteration k satisfying (3.7) and under the assumptions of Theorem 3.3. Hence, there exists an iteration $k \leq n$ for which the error bound (3.8) holds. To further simplify the expression (3.8), we need to bound $\sigma_{\min}(M_R\widetilde{Z}_k)$; we have using (3.7), (C.4), (C.5), and (C.12),

$$\begin{aligned}\sigma_{\min}(M_R\widetilde{Z}_k) &\geq \sigma_{\min}(\widehat{V}_k) - \|M_R\Delta Z_k^{(1)}\|_F - \|M_R\Delta Z_k^{(2)}\|_F \\ &\gtrsim 3/4 - c(n,k)\left(u_m\eta_R\kappa(M_R) + u_a\kappa(AM_R^{-1})\kappa(M_R)\right).\end{aligned} \tag{C.15}$$

Under assumption (3.16) ensuring $u_m\eta_R\kappa(M_R) \ll 1$ and $u_a\kappa(AM_R^{-1})\kappa(M_R) \ll 1$, we guarantee $\sigma_{\min}(M_R\widetilde{Z}_k) \gtrsim 3/4$. Dropping second order terms in the bounds (C.14) and (C.15), the expressions

of βu_g and $\lambda\varepsilon_x$ in (3.8) become

$$\beta u_g = u_g \frac{\kappa(M_R)}{\sigma_{\min}(M_R\widetilde{Z}_k)} \frac{\|A\widetilde{Z}_k\|_F}{\|AM_R^{-1}\|_F} \lesssim c(n,k)u_g\kappa(M_R),$$

$$\lambda\varepsilon_x = c(n,k)\left(u_m\eta_R\kappa(M_R) + u_g\kappa(M_R) + u_a\kappa(A)\rho_A^R\right)\kappa(AM_R^{-1})^{-1},$$

which leads to (3.17).

D. Proof of Theorem 3.4

Most of the proof stays identical to the proof of Theorem 3.3. In particular, the relations (C.2), (C.5), (C.7), (C.8), and (C.10) obtained for right-preconditioned GMRES are still valid and condition (3.5) is met for the same reasons.

The differences concern the parts linked to the computation of the solution approximation carried by the model (3.4). In the case of flexible-preconditioning, the basis $\widehat{Z}_k$ is stored explicitly, and so the solution approximation is obtained directly from a standard matrix–vector product with $\widehat{Z}_k$ in precision u_g. We have

$$\widehat{x}_k = \mathrm{fl}(\widehat{Z}_k\widehat{y}_k) = (\widehat{Z}_k + \Delta Z_k^{(3)})\widehat{y}_k, \qquad |\Delta Z_k^{(3)}| \lesssim c(n,k)u_g|\widehat{Z}_k|, \tag{D.1}$$

which, accounting for (C.5), leads to

$$\widehat{x}_k = \widetilde{Z}_k\widehat{y}_k + \Delta_x, \qquad \Delta_x = \left(\Delta Z_k^{(3)} - \Delta Z_k^{(2)}\right)\widehat{y}_k.$$

Using (C.7) and (C.8) to bound $\|\Delta Z_k^{(2)}\widehat{y}_k\|_2$, and using $\|\widehat{Z}_k\|_F \lesssim k^{1/2}\|M_R^{-1}\|_F$ combined with (C.10) to bound $\|\Delta Z_k^{(3)}\widehat{y}_k\|_2$, we have

$$\|\Delta_x\|_2 \lesssim c(n,k)\left(u_g\kappa(M_R) + u_a\kappa(A)\rho_A^R\right)\|\widetilde{Z}_k\widehat{y}_k\|_2. \tag{D.2}$$

Thus, comparing the previous bound on $\|\Delta_x\|_2$ with (C.11), we can observe that without the requirement to reapply the preconditioner M_R^{-1}, flexible-preconditioning can spare the term $u_m\eta_R\kappa(M_R)$ and remove the dependence on the precision u_m. From (D.2), we identify the accuracy parameter $\varepsilon_x \equiv c(n,k)(u_g\kappa(M_R) + u_a\kappa(A)\rho_A^R)$ in the model (3.4) which, under assumption (3.20), satisfies $\varepsilon_x \ll 1$ such that condition (3.6) is met.

We can now rework the forward error bound with the new value ε_x. Identically to the proof of Theorem 3.3, we write

$$\lambda\varepsilon_x = c(n,k)\left(u_g\kappa(M_R) + u_a\kappa(A)\rho_A^R\right)\kappa(AM_R^{-1})^{-1}$$

in (3.8) and we recover the forward error bound (3.21).

References

1. E. Agullo, F. Cappello, S. Di, L. Giraud, X. Liang, and N. Schenkels, *Exploring variable accuracy storage through lossy compression techniques in numerical linear algebra: a first application to flexible GMRES*, Research Report RR-9342, Inria Bordeaux Sud-Ouest, May 2020.

2. J. I. Aliaga, H. Anzt, T. Grützmacher, E. S. Quintana-Ortí, and A. E. Tomás, *Compressed basis GMRES on high-performance graphics processing units*, Int. J. High Performance Computing Applications, (2022), p. 109434202211151, https://doi.org/10.1177/10943420221115140.
3. P. Amestoy, A. Buttari, N. J. Higham, J.-Y. L'Excellent, T. Mary, and B. Vieublé, *Combining Sparse Approximate Factorizations with Mixed-precision Iterative Refinement*, ACM Trans. Math. Software, 49 (2023), p. 1–29, https://doi.org/10.1145/3582493.
4. P. Amestoy, A. Buttari, N. J. Higham, J.-Y. L'Excellent, T. Mary, and B. Vieublé, *Five-Precision GMRES-Based Iterative Refinement*, SIAM J. Matrix Anal. Appl., 45 (2024), p. 529–552, https://doi.org/10.1137/23m1549079.
5. H. Anzt, J. Dongarra, G. Flegar, N. J. Higham, and E. S. Quintana-Ortí, *Adaptive precision in block-Jacobi preconditioning for iterative sparse linear system solvers*, Concurrency Computat.: Pract. Exper., 31 (2019), p. e4460, https://doi.org/10.1002/cpe.4460.
6. H. Anzt, V. Heuveline, and B. Rocker, *Mixed Precision Iterative Refinement Methods for Linear Systems: Convergence Analysis Based on Krylov Subspace Methods*, in Applied Parallel and Scientific Computing, Springer Berlin Heidelberg, 2012, pp. 237–247, https://doi.org/10.1007/978-3-642-28145-7_24.
7. M. Arioli and I. S. Duff, *Using FGMRES to obtain backward stability in mixed precision*, Electron. Trans. Numer. Anal., 33 (2008), pp. 31–44.
8. M. Arioli, I. S. Duff, S. Gratton, and S. Pralet, *A Note on GMRES Preconditioned by a Perturbed LDL^T Decomposition with Static Pivoting*, SIAM J. Sci. Comput., 29 (2007), pp. 2024–2044, https://doi.org/10.1137/060661545.
9. C. Boutsikas, P. Drineas, and I. C. F. Ipsen, *Small Singular Values Can Increase in Lower Precision*, SIAM J. Matrix Anal. Appl., 45 (2024), p. 1518–1540, https://doi.org/10.1137/23m1557209.
10. A. Buttari, J. Dongarra, J. Kurzak, P. Luszczek, and S. Tomov, *Using Mixed Precision for Sparse Matrix Computations to Enhance the Performance while Achieving 64-bit Accuracy*, ACM Trans. Math. Software, 34 (2008), pp. 1–22, https://doi.org/10.1145/1377596.1377597.
11. A. Buttari, N. J. Higham, T. Mary, and B. Vieublé, *A modular framework for the backward error analysis of GMRES*. working paper or preprint, Mar. 2024.
12. E. Carson and I. Daužickaite, *Mixed precision sketching for least-squares problems and its application in GMRES-based iterative refinement*. working paper or preprint, 2024, https://arxiv.org/abs/2410.06319.
13. E. Carson and I. Daužickaite, *The stability of split-preconditioned FGMRES in four precisions*, Electron. Trans. Numer. Anal., 60 (2024), p. 40–58, https://doi.org/10.1553/etna_vol60s40.
14. E. Carson and N. J. Higham, *A New Analysis of Iterative Refinement and Its Application to Accurate Solution of Ill-Conditioned Sparse Linear Systems*, SIAM J. Sci. Comput., 39 (2017), pp. A2834–A2856, https://doi.org/10.1137/17m1122918.
15. E. Carson and N. J. Higham, *Accelerating the Solution of Linear Systems by Iterative Refinement in Three Precisions*, SIAM J. Sci. Comput., 40 (2018), pp. A817–A847, https://doi.org/10.1137/17M1140819.
16. E. Carson, N. J. Higham, and S. Pranesh, *Three-Precision GMRES-Based Iterative Refinement for Least Squares Problems*, SIAM J. Sci. Comput., 42 (2020), pp. A4063–A4083, https://doi.org/10.1137/20m1316822.
17. E. Carson and N. Khan, *Mixed Precision Iterative Refinement with Sparse Approximate Inverse Preconditioning*, SIAM J. Sci. Comput., 45 (2023), pp. C131–C153, https://doi.org/10.1137/22m1487709.
18. T. A. Davis and Y. Hu, *The university of Florida sparse matrix collection*, ACM Trans. Math. Software, 38 (2011), pp. 1–25, https://doi.org/10.1145/2049662.2049663.
19. L. Giraud, S. Gratton, and J. Langou, *Convergence in Backward Error of Relaxed GMRES*, SIAM J. Sci. Comput., 29 (2007), p. 710–728, https://doi.org/10.1137/040608416, http://dx.doi.org/10.1137/040608416.
20. S. Graillat, F. Jézéquel, T. Mary, and R. Molina, *Adaptive Precision Sparse Matrix–Vector Product and Its Application to Krylov Solvers*, SIAM J. Sci. Comput., 46 (2024), p. C30–C56, https://doi.org/10.1137/22m1522619.
21. S. Gratton, E. Simon, D. Titley-Peloquin, and P. L. Toint, *A Note on Inexact Inner Products*

in GMRES, SIAM J. Matrix Anal. Appl., 43 (2022), p. 1406–1422, https://doi.org/10.1137/20m1320018, http://dx.doi.org/10.1137/20M1320018.
22. A. Greenbaum, V. Pták, and Z. Strakoš, *Any nonincreasing convergence curve is possible for GMRES*, SIAM J. Matrix Anal. Appl., 17 (1996), pp. 465–469, https://doi.org/10.1137/s0895479894275030.
23. N. J. Higham, *Accuracy and Stability of Numerical Algorithms*, Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, second ed., 2002, https://doi.org/10.1137/1.9780898718027.
24. N. J. Higham and T. Mary, *A New Approach to Probabilistic Rounding Error Analysis*, SIAM J. Sci. Comput., 41 (2019), pp. A2815–A2835, https://doi.org/10.1137/18M1226312.
25. J. D. Hogg and J. A. Scott, *A fast and robust mixed-precision solver for the solution of sparse symmetric linear systems*, ACM Trans. Math. Software, 37 (2010), p. 1–24, https://doi.org/10.1145/1731022.1731027.
26. N. Lindquist, P. Luszczek, and J. Dongarra, *Accelerating Restarted GMRES With Mixed Precision Arithmetic*, IEEE Trans. Parallel Distributed Sys., 33 (2022), pp. 1027–1037, https://doi.org/10.1109/tpds.2021.3090757.
27. Q. Liu, R. B. Morgan, and W. Wilcox, *Polynomial Preconditioned GMRES and GMRES-DR*, SIAM J. Sci. Comput., 37 (2015), p. S407–S428, https://doi.org/10.1137/140968276, http://dx.doi.org/10.1137/140968276.
28. J. A. Loe, C. A. Glusa, I. Yamazaki, E. G. Boman, and S. Rajamanickam, *Experimental Evaluation of Multiprecision Strategies for GMRES on GPUs*, in 2021 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW), IEEE, June 2021, p. 469–478, https://doi.org/10.1109/ipdpsw52791.2021.00078.
29. E. Oktay and E. Carson, *Multistage mixed precision iterative refinement*, Numer. Linear Algebra Appl., (2022), https://doi.org/10.1002/nla.2434.
30. C. C. Paige, M. Rozložník, and Z. Strakoš, *Modified Gram-Schmidt (MGS), Least Squares, and Backward Stability of MGS-GMRES*, SIAM J. Matrix Anal. Appl., 28 (2006), pp. 264–284, https://doi.org/10.1137/050630416.
31. Y. Saad, *A flexible inner-outer preconditioned GMRES algorithm*, SIAM J. Sci. Comput., 14 (1993), pp. 461–469, https://doi.org/10.1137/0914028.
32. 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, https://doi.org/10.1137/0907058.
33. V. Simoncini and D. B. Szyld, *Theory of Inexact Krylov Subspace Methods and Applications to Scientific Computing*, SIAM J. Sci. Comput., 25 (2003), p. 454–477, https://doi.org/10.1137/s1064827502406415, http://dx.doi.org/10.1137/S1064827502406415.
34. K. Turner and H. F. Walker, *Efficient High Accuracy Solutions with GMRES(m)*, SIAM J. Sci. Statist. Comput., 13 (1992), pp. 815–825, https://doi.org/10.1137/0913048.
35. B. Vieublé, *Mixed precision iterative refinement for the solution of large sparse linear systems*, PhD thesis, INP Toulouse, Nov. 2022.
36. A. J. Wathen, *Preconditioning*, Acta Numerica, 24 (2015), pp. 329–376, https://doi.org/10.1017/s0962492915000021.
37. I. Yamazaki, S. Tomov, T. Dong, and J. Dongarra, *Mixed-precision orthogonalization scheme and adaptive step size for improving the stability and performance of CA-GMRES on GPUs*, in Lecture Notes in Computer Science, Springer International Publishing, 2015, pp. 17–30, https://doi.org/10.1007/978-3-319-17353-5_2.