Five-Precision GMRES-based iterative refinement

Patrick Amestoy, Alfredo Buttari, Nicholas Higham, Jean-Yves L'excellent, Théo Mary, Bastien Vieublé

FIVE-PRECISION GMRES-BASED ITERATIVE REFINEMENT *

PATRICK AMESTOY†, ALFREDO BUTTARI‡, NICHOLAS J. HIGHAM§, JEAN-YVES L'EXCELLENT†, THEO MARY¶, AND BASTIEN VIEUBLÉ§

Abstract. GMRES-based iterative refinement in three precisions (GMRES-IR3), proposed by Carson and Higham in 2018, uses a low precision LU factorization to accelerate the solution of a linear system without compromising numerical stability or robustness. GMRES-IR3 solves the update equation of iterative refinement using GMRES preconditioned by the LU factors, where all operations within GMRES are carried out in the working precision u, except for the matrix-vector products and the application of the preconditioner, which require the use of extra precision u^2. The use of extra precision can be expensive, and is especially unattractive if it is not available in hardware; for this reason, existing implementations have not used extra precision, despite the absence of an error analysis for this approach. In this article, we propose to relax the requirements on the precisions used within GMRES, allowing the use of arbitrary precisions u_p for applying the preconditioned matrix–vector product and u_g for the rest of the operations. We obtain the five-precision GMRES-based iterative refinement (GMRES-IR5) algorithm which has the potential to solve relatively badly conditioned problems in less time and memory than GMRES-IR3. We develop a rounding error analysis that generalizes that of GMRES-IR3, obtaining conditions under which the forward and backward errors converge to their limiting values. Our analysis makes use of a new result on the backward stability of MGS-GMRES in two precisions. On hardware where three or more arithmetics are available, which is becoming very common, the number of possible combinations of precisions in GMRES-IR5 is extremely large. We provide an analysis of our theoretical results that identifies a relatively small subset of relevant combinations. By choosing from within this subset one can achieve different levels of tradeoff between cost and robustness, which allows for a finer choice of precisions depending on the problem difficulty and the available hardware. We carry out numerical experiments on random dense and SuiteSparse matrices to validate our theoretical analysis and discuss the complexity of GMRES-IR5.

Key words. iterative refinement, GMRES, linear system, mixed precision, multiple precision, rounding error analysis, floating-point arithmetic, backward error, forward error, preconditioning

AMS subject classifications. 65G50, 65F05, 65F08, 65F50, 65F10

1. Introduction.

Modern hardware increasingly supports low precision floating-point arithmetics that provide unprecedented speed, communication, and energy benefits. This has generated renewed interest in mixed precision algorithms that combine these lower precision arithmetics with higher precision ones to achieve both high performance and high accuracy [19]. Among such mixed precision algorithms, iterative refinement for the solution of a linear system $Ax = b$ is one of the oldest and most successful [3], [14, Chap. 12], [25], [30].

We recall, in Algorithm 1.1, the iterative refinement recently proposed and analyzed by Carson and Higham [5]. Algorithm 1.1 includes and generalizes previous iterative refinement algorithms in two ways. First, each step of the i^{th} iteration of the algorithm may be carried out in a different precision: the initial LU factorization

*Version of November 1, 2023.

Funding: The work of the third author was supported by Engineering and Physical Sciences Research Council grant EP/P020720/1, the Royal Society, and the Exascale Computing Project (17-SC-20-SC), a collaborative effort of the U.S. Department of Energy Office of Science and the National Nuclear Security Administration..

†Mumps Technologies, ENS Lyon, 46 Allée d'Italie, F-69007 Lyon, France (patrick.amestoy@mumps-tech.com; jean-yves.l.excellent@mumps-tech.com)

‡CNRS, IRIT, 2 Rue Charles Camichel, F-31071 Toulouse, France (alfredo.buttari@irit.fr)

§Department of Mathematics, The University of Manchester, Manchester, M13 9PL, UK (nick.higham@manchester.ac.uk; bastien.vieuble@manchester.ac.uk)

¶Sorbonne Université, CNRS, LIP6, Paris, F-75005, France (theo.mary@lip6.fr)

Algorithm 1.1 Iterative refinement

Input: an $n \times n$ matrix A and a right-hand side b.
Output: an approximate solution to $Ax = b$.

1: Compute the LU factorization $A = LU$. u_f
2: Initialize x_0 (to, e.g., $U^{-1}L^{-1}b$). u_f
3: **for** $i = 0, \ldots,$ converged **do**
4: Compute $r_i = b - Ax_i$. u_r
5: Solve $Ad_i = r_i$. u_s
6: Compute $x_{i+1} = x_i + d_i$. u
7: **end for**

of A is performed in precision u_f, the residual $r_i = b - Ax_i$ is computed in precision u_r, and the solution x_i is updated in the target, working precision u. Second, the correction term d_i is obtained by solving the system $Ad_i = r_i$ with a generic solver assumed to compute d_i with relative accuracy of order $u_s \geq u$. The precision u_s is a measure of the quality of the solution the solver produces rather than an input to the solver.

The traditional choice of solver, which we call the LU solver, is to compute $d_i = U^{-1}L^{-1}r_i$ by substitution in precision u_f, so that d_i is computed at accuracy $u_s = O(u_f)$. In this case, Algorithm 1.1 employs up to three precisions. We denote this variant by LU-IR3. While LU-IR3 can be very attractive for well-conditioned matrices, it is only guaranteed to converge when $\kappa(A)u_f \lll 1$, where $\kappa(A) = \|A\|\|A^{-1}\|$ denotes the condition number of A and "$\lll$" means "is sufficiently less than". This condition can be quite restrictive, especially when low precision arithmetic is used for the factorization: for example, the condition becomes $\kappa(A) \lll 2 \times 10^3$ with IEEE fp16 (half) precision, and $\kappa(A) \lll 3 \times 10^2$ with bfloat16.

Carson and Higham [4] take $u_f = u$ and solve $Ad_i = r_i$ by GMRES preconditioned by the computed LU factors of A, with the preconditioner and the matrix–vector product with A applied in higher precision u^2 and the residual also computed at higher precision, $u_r = u^2$. Their motivation was to compute solutions with forward error of order u even when $\kappa(A)$ is of order u^{-1}. Carson and Higham [5] subsequently generalized this algorithm in order to be able to exploit lower precision arithmetic for the factorization. This algorithm, called GMRES-based iterative refinement in three precisions (GMRES-IR3), is recalled in Algorithm 1.2. On line 5, $\widetilde{A}$ is not explicitly formed, but its action on a vector in the Arnoldi process of the GMRES algorithm is obtained with a matrix multiplication followed by a forward substitution and a backward substitution corresponding to the L and U factors, respectively.

GMRES-IR3 is shown in [5] to be guaranteed to converge as long as $\kappa(A)^2 u_f^2 u \lll 1$, and it is therefore able to handle much more ill-conditioned matrices than LU-IR3. However, the requirement that the preconditioned matrix–vector product be applied in precision u^2 is a practical limitation, because this can be expensive; it is particularly inconvenient if the target accuracy u is double precision, as it requires applying the preconditioner and the matrix–vector product with A in quadruple precision, an arithmetic that is not natively supported on most modern hardware and is an order of magnitude slower than double precision when implemented in software [15]. In fact, practical implementations of GMRES-IR3, as developed in [1], [10], [11], [12] and implemented in the MAGMA library [24] and the NVIDIA cuSOLVER library [8], have relaxed this requirement by applying the preconditioned matrix–vector product

Algorithm 1.2 GMRES based iterative refinement in three precisions (GMRES-IR3)

Input: an $n \times n$ matrix A and a right-hand side b.
Output: Approximate solution $\widehat{x}$ to $Ax = b$

1: Compute the LU factorization $A = LU$. u_f
2: Initialize x_0 (e.g., to $U^{-1}L^{-1}b$). u_f
3: **for** $i = 0, \ldots,$ converged **do**
4: Compute $r_i = b - Ax_i$. u_r
5: Solve $U^{-1}L^{-1}Ad_i = U^{-1}L^{-1}r_i$ by GMRES at precision u with matrix–vector products with $\widetilde{A} = U^{-1}L^{-1}A$ computed at precision u^2.
6: Compute $x_{i+1} = x_i + d_i$. u
7: **end for**

in double rather than quadruple precision—even though the error analysis of [5] does not cover this case. GMRES-IR variants for symmetric positive definite systems and least squares problems have also allowed just two precisions to be used [6], [20]. The question arises of whether we can use a lower precision to compute the preconditioned matrix–vector product within GMRES and still obtain a GMRES-IR solver able to handle more ill-conditioned matrices than LU-IR.

These practical constraints and theoretical questions lead us to propose new variants of GMRES-based iterative refinement with relaxed requirements on the precisions used within the GMRES solver. We allow the preconditioner (the LU factors) and the matrix–vector product with A to be applied in an arbitrary precision u_p, with $u_p \geq u^2$. We also allow the rest of the GMRES computations to be performed in an arbitrary precision u_g, with $u_g \geq u$. We obtain Algorithm 3.1, which has up to *five* independent precisions in play and which we thus call GMRES-IR5. We generalize the rounding error analysis of Carson and Higham [4], [5] to this new GMRES-IR5 variant and obtain several novel combinations of precision parameters that achieve different levels of tradeoff between cost and robustness. We summarize the conclusions of this analysis in Theorems 3.1 and 3.2 which give, respectively, a bound on the backward error of MGS-GMRES using an arbitrary matrix–vector product and conditions under which the forward error and backward error of GMRES-IR5 achieve certain limiting values. GMRES-IR5 therefore fills the gap between LU-IR3 and GMRES-IR3, allowing for new variants that can be much faster than GMRES-IR3 but that remain more robust than LU-IR3. When multiple arithmetics are available (for example, current supercomputers commonly support bfloat16, fp16, fp32 and fp64), the number of possible combinations is extremely large; based on our analysis and the resulting error bounds, we provide recipes to trim the space of possible combinations down to a relatively small subset of relevant ones. These include combinations where $u_p = u$ and, therefore, our results provide convergence guarantees for the existing implementations mentioned above.

Note that even though GMRES-IR5 has five precisions parameters in play, this does not mean that it uses five *different* arithmetics at the same time, since some of the precisions can be equal. In fact, most of the relevant combinations of precisions identified by our analysis use only two or three different precisions. Nevertheless, we discover some meaningful variants that employ, for the first time, four or even five different precisions in the same refinement step. (We specify "in the same refinement step" because dynamically changing the precisions from one iteration to another could already lead to such a number of different precisions across all steps, and even more

in the case of arbitrary precision refinement [22], [23]).

The rest of this article is organized as follows. We begin, in section 2, by providing technical background on LU-IR3 and GMRES-IR3. In section 3, we present the generalized error analysis of GMRES-IR5 in five precisions and its different combinations. In section 4, we support our analysis with numerical experiments on random dense matrices and matrices from the SuiteSparse collection [**?**]. In section 5, we discuss the complexity of GMRES-IR5. We provide our concluding remarks in section 6.

2. Preliminaries on LU-IR3 and GMRES-IR3. In this section we summarize our notation and assumptions for the analysis, and briefly recall the essential technical background on the LU-IR3 and GMRES-IR3 algorithms of Carson and Higham [4, 5].

We use the standard model of floating-point arithmetic [14, sect. 2.2] and we use the notation $\mathrm{fl}(\cdot)$ to denote the computed value of a given expression. For any integer k we define

$$\gamma_k = \frac{ku}{1-ku}.$$

A superscript on γ denotes that u carries that superscript as a subscript; thus $\gamma_k^f = ku_f/(1-ku_f)$, for example. We also use the notation $\widetilde{\gamma}_k = \gamma_{ck}$ to hide modest constants c.

The error bounds obtained by our analysis depend on the problem dimension n, the maximum number of iterations k of GMRES over all iterative refinement steps, and the growth factor ρ_n of the LU factorization. In parts of the analysis, we will gather these constants into a generic function $f(n,k,\rho_n)$. The constants depending on n are known to be pessimistic [7], [17], [18] and, with standard pivoting strategies, ρ_n is almost always small in practice and of order a constant [14, chap. 9] (for exceptions, see [13]). Therefore, for the sake of the readability, we do not always keep track of the precise value of $f(n,k,\rho_n)$. When we drop constants $f(n,k,\rho_n)$ from an inequality we write the inequality using "$\lll$". A convergence condition expressed as "$\kappa(A) \lll \theta$" can be read as "$\kappa(A)$ is sufficiently less than θ". Finally, we also use the notation $\lesssim$ and $\approx$ when dropping negligible second order terms in the error bounds; we note that what makes a second-order term negligible depends on the local context: for example, the term u_f^2 is not necessarily negligible in the expression $u + u_f^2$, but can be safely dropped from the expression $u_f + u_f^2$.

We will also make use of 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 consider linear systems $Ax = b$, where $A \in \mathbb{R}^{n\times n}$ is nonsingular and $b \in \mathbb{R}^n$. The forward error of an approximation $\widehat{x}$ is $\|x-\widehat{x}\|/\|x\|$, while the (normwise) backward error of $\widehat{x}$ is [14, sec. 7.1]

$$\min\{\,\epsilon : (A+\Delta A)\widehat{x} = b+\Delta b,\ \|\Delta A\| \le \epsilon\|A\|,\ \|\Delta b\| \le \epsilon\|b\|\,\} = \frac{\|b-A\widehat{x}\|}{\|A\|\,\|\widehat{x}\|+\|b\|}.$$

We define the componentwise condition numbers

$$\mathrm{cond}(A,x) = \frac{\||A^{-1}||A||x|\|}{\|x\|}, \quad \mathrm{cond}(A) = \||A^{-1}||A|\|,$$

where $|A| = (|a_{ij}|)$.

Our error analysis uses both the ∞-norm and the Frobenius norm, denoted by $\|\cdot\|_\infty$ and $\|\cdot\|_F$, respectively, and we write $\kappa_\infty(A)$ and $\kappa_F(A)$ for the corresponding

condition numbers of A. We will use unsubscripted norms or condition numbers when the constants depending on the problem dimensions have been dropped, since the norms are equivalent.

We denote by q the maximum number of nonzeros in any row of $[A\,b]$; thus $q = n + 1$ for a dense matrix A and a vector b. As in [5], however, we do not make any assumptions on the sparsity of the LU factors L and U.

Carson and Higham [5] perform the error analysis of Algorithm 1.1 with a general solver used at step 5. The solver is assumed to satisfy the conditions

$$\widehat{d}_i = (I + u_s E_i)d_i, \quad u_s\|E_i\|_\infty < 1, \tag{2.1a}$$

$$\|\widehat{r}_i - A\widehat{d}_i\|_\infty \le u_s(c_1\|A\|_\infty\|\widehat{d}_i\|_\infty + c_2\|\widehat{r}_i\|_\infty), \tag{2.1b}$$

where $\widehat{d}_i$ and $\widehat{r}_i$ denote the computed d_i and r_i, and where E_i, c_1, and c_2, are functions of n, A, $\widehat{r}_i$, and u_s. These conditions are needed for the normwise forward and backward error analyses, respectively.

In analyzing iterative refinement we aim to show that the forward error and backward error decrease until they reach a certain size called the limiting forward error or backward error (sometimes also called the attainable accuracy). We will informally refer to attainment of this level as "convergence", while recognizing that the error does not necessarily converge in the formal sense.

Corollary 3.3 of [5] shows that as long as

$$u_s\|E_i\| \lll 1 \tag{2.2}$$

for each i, the forward error converges to a limiting value of $qu_r\,\mathrm{cond}(A, x) + u$. Here, we have simplified the convergence condition by ignoring a term that is shown in [5] to be dominated in practice by the left-hand side of (2.2). Corollary 4.2 of [5] shows that if

$$u_s(c_1\kappa(A) + c_2) \lll 1, \tag{2.3}$$

then the backward error converges to a limiting value of $qu_r + u$. The convergence conditions of Algorithm 1.1 therefore depend only on u_s, whereas the limiting accuracy and backward errors depend on u and u_r.

This analysis of Algorithm 1.1 can be used to derive convergence conditions when we specialize the solver at step 5 if this solver meets the assumptions (2.1a) and (2.1b). To do so we only need to compute the left-hand side quantities of (2.2) and (2.3) which are respectively upper bounds on the forward and backward errors of the solver.

In the case of LU-IR3, we have $u_s \equiv u_f$, $\|E_i\|_\infty \equiv f(n, \rho_n)\kappa_\infty(A)$, and $c_1 = c_2 = f(n, \rho_n)$, so the convergence condition is

$$\kappa(A) \lll u_f^{-1}, \tag{2.4}$$

for both the forward and backward errors.

In the case of GMRES-IR3, that is, Algorithm 1.2, we have instead $u_s \equiv u$ and $\|E_i\|_\infty \equiv f(n, k, \rho_n)\kappa_\infty(\widetilde{A})$ [4, sec. 3], where $\widetilde{A} = \widehat{U}^{-1}\widehat{L}^{-1}A$ is the matrix preconditioned by the computed LU factors $\widehat{L}$ and $\widehat{U}$. Using the bound

$$\kappa_\infty(\widetilde{A}) \lesssim (1 + \gamma_n^f\kappa_\infty(A))^2 \tag{2.5}$$

from [5, Eq. (8.3)], which shows that $\widetilde{A}$ is usually better conditioned than A, (2.2) yields the condition

$$\kappa(A) \lll u^{-1/2}u_f^{-1}, \tag{2.6}$$

for the forward error to converge. This is a significantly less restrictive condition than (2.4) for LU-IR3. Similarly, for GMRES-IR3 we can show that $c_1 = f(n,k,\rho_n)\|\widetilde{A}\|_\infty$ and $c_2 = f(n,k,\rho_n)\kappa_\infty(A)$, so that after bounding $\|\widetilde{A}\|_\infty$ with [4, Eq. (3.2)] the convergence condition (2.3) for the backward error becomes

$$\kappa(A) \lll u^{-1/2}u_f^{-1/2}, \tag{2.7}$$

which is also less restrictive than (2.4) when $u < u_f$. Note that the original condition for the backward error proposed in [5] was $\kappa(A)u \lll 1$ and is substantially less restrictive than (2.7). The reason for this discrepancy is that in [5] the assumption is made that c_1 in (2.3) is of order $f(n,k,\rho_n)$, which is too optimistic in general.

3. Error analysis of GMRES-IR5. In this section we generalize the error analysis of GMRES-IR3 (Algorithm 1.2) [4], [5] to GMRES-IR5, which is defined in Algorithm 3.1. First, we extend in section 3.1 the analysis of Paige, Rozložník, and Strakoš [27] on the backward stability of MGS-GMRES to arbitrary matrix–vector products satisfying a generic error bound. Second, we use this generalized analysis in section 3.2 to bound the forward and backward errors of the two precision MGS-GMRES solver used at step 5 for the resolution of $\widetilde{A}d_i = s_i$, where $\widetilde{A} = \widehat{U}^{-1}\widehat{L}^{-1}A$ and $s_i = \widehat{U}^{-1}\widehat{L}^{-1}r_i$. We then use these bounds in section 3.3 to rewrite (2.2) and (2.3) to obtain the specialized conditions on $\kappa(A)$ for GMRES-IR5 to converge (note that the limiting backward error and forward error are left unchanged because they only depend on u and u_r). Finally, we use these convergence conditions in section 3.4 to identify which combinations of precisions are meaningful.

Algorithm 3.1 GMRES based iterative refinement in five precisions (GMRES-IR5)

Input: an $n \times n$ matrix A and a right-hand side b.
Output: Approximate solution $\widehat{x}$ to $Ax = b$

1: Compute the LU factorization $A = LU$. u_f
2: Initialize x_0 (to, e.g., $U^{-1}L^{-1}b$). u_f
3: **for** $i = 0, \ldots,$ converged **do**
4: Compute $r_i = b - Ax_i$. u_r
5: Solve $U^{-1}L^{-1}Ad_i = U^{-1}L^{-1}r_i$ by GMRES at precision u_g with matrix–vector products with $\widetilde{A} = U^{-1}L^{-1}A$ computed at precision u_p.
6: Compute $x_{i+1} = x_i + d_i$. u
7: **end for**

Our analysis makes use of the following three assumptions on the precisions:

- $u_g \geq u$: since the solution computed by GMRES is stored in the working precision u, we do not expect running GMRES in precision $u_g < u$ to give a significant benefit.
- $\kappa(A)u < 1$: this assumption is already present in the three-precision analysis [5]; here, we use it for various discussions, but it is not required to derive our new convergence conditions.
- $\kappa(A)u_p < 1$: this assumption is needed to drop certain second order terms in our analysis. It guarantees that $\kappa(A)^2u_p^2$ is negligible relative to $\kappa(A)u_p$.

In addition, as in [5], we also assume that the growth factor ρ_n of the LU factorization is of modest size, which is usually the case when common pivoting strategies are used (e.g., partial pivoting).

3.1. Error analysis of MGS-GMRES with arbitrary matrix–vector products. We assume that the MGS-GMRES variant of GMRES is used in GMRES-IR5, so we need to bound the backward error of two-precision MGS-GMRES for the solution of the preconditioned system $\widetilde{A}d_i = s_i$. The analysis of Paige et al. [27] is for fixed precision, unpreconditioned MGS-GMRES, and therefore is not directly applicable. Note that in the analysis of [4] the preconditioner is applied in precision u^2, and using the assumption $\kappa(A)u < 1$ it is shown that the products with $\widetilde{A}$ in precision u^2 are at least as accurate as the products with A in precision u, so that the backward stability result of [27] still holds. In our case, the same argument does not apply, and so we must generalize the backward stability result [27, Eq. (8.15)] to the case of arbitrary matrix–vector products satisfying a generic error bound. We state the conclusion of our analysis in the next theorem.

THEOREM 3.1. *Consider the solution of a linear system*

$$Bx = d, \quad B \in \mathbb{R}^{n\times n}, \quad 0 \neq d \in \mathbb{R}^n \tag{3.1}$$

with an MGS-GMRES solver carrying out its operations in precision u_g, except for the products with B, which satisfy instead

$$\mathrm{fl}(Bv) = Bv + f, \quad \|f\|_2 \lesssim \epsilon_p \|B\|_F \|v\|_2, \tag{3.2}$$

where $\epsilon_p > 0$ is a parameter quantifying the stability of the matrix–vector products. Provided that

$$\sigma_{\min}(B) \gtrsim (k^{1/2}\epsilon_p + \widetilde{\gamma}_{kn}^g)\|B\|_F, \tag{3.3}$$

there is a step $k \leq n$ such that the algorithm produces a computed $\widehat{x}_k$ satisfying

$$(B + \Delta B)\widehat{x}_k = d + \Delta d, \tag{3.4a}$$

$$\|\Delta B\|_F \lesssim (k^{1/2}\epsilon_p + \widetilde{\gamma}_{kn}^g)\|B\|_F, \tag{3.4b}$$

$$\|\Delta d\|_2 \lesssim \widetilde{\gamma}_{kn}^g \|d\|_2. \tag{3.4c}$$

Proof. The proof of Theorem 3.1 relies on the analysis of [27] and, more precisely, on [27, sec. 8] and [27, Eq. (4.3)] therein. For the sake of readability, we will not attempt to make this proof self-contained but, rather, we will highlight the differences with the analysis in [27] and refer the reader to the appendix of [29] for the full details. The original notation has been slightly adapted to be consistent with the notation of this article inherited from [4] and [5].

In our version of MGS-GMRES we consider a product with B satisfying (3.2). We now show that considering (3.2) with $\epsilon_p \neq \gamma_n^g$ mainly changes [27, Eq. (4.3)]. Let us consider $\widehat{V}_k = [\widehat{v}_1, \ldots, \widehat{v}_k] \in \mathbb{R}^{n\times k}$, the matrix of computed basis vectors, and $\dot{V}_k = [\dot{v}_1, \ldots, \dot{v}_k]$ the same matrix but with its columns correctly normalized; that is, for $j \leq k$,

$$\begin{aligned} \widehat{v}_j &= \dot{v}_j + \Delta v_j^{(1)}, \quad \|\Delta v_j^{(1)}\|_2 \leq \widetilde{\gamma}_n^g, \\ \widehat{V}_k &= \dot{V}_k + \Delta V_k^{(1)}, \quad \Delta V_k^{(1)} = [\Delta v_1^{(1)}, \ldots, \Delta v_k^{(1)}], \end{aligned} \tag{3.5}$$

where $\Delta v_j^{(1)}$ is the error for the normalization of $\widehat{v}_j$ and $\Delta V_k^{(1)}$ is the accumulated error for the normalization of the basis at step k. By (3.2) and (3.5), we obtain

$$\begin{aligned} \mathrm{fl}(B\widehat{v}_j) &= B(\dot{v}_j + \Delta v_j^{(1)}) + f_j \\ &= B\dot{v}_j + \Delta v_j^{(2)}, \end{aligned}$$

where $\Delta v_j^{(2)} = B\Delta v_j^{(1)} + f_j$ satisfies $\|\Delta v_j^{(2)}\|_2 \lesssim (\epsilon_p + \widetilde{\gamma}_n^g)\|B\|_F$ since $\|\dot{v}_j\|_2 = 1$ and $\|f_j\|_2 \lesssim \epsilon_p\|B\|_F\|\dot{v}_j + \Delta v_j^{(1)}\|_2$. We therefore obtain

$$\mathrm{fl}(B\widehat{V}_k) = B\dot{V}_k + \Delta V_k^{(2)}, \qquad \|\Delta V_k^{(2)}\|_F \lesssim k^{1/2}(\epsilon_p + \widetilde{\gamma}_n^g)\|B\|_F, \tag{3.6}$$

where $\Delta V_k^{(2)}$ contains the error for both the product and the normalization at the kth iteration. Equation (3.6) is our new version of [27, Eq. (4.3)]; adapting the remainder of [27, sec. 8] to take this change into account is straightforward. Consequently, we show that at the $(\bar{m}-1)$st iteration, MGS-GMRES has computed a backward stable solution of the system, where $\bar{m}$ satisfies [27, Eq. (6.1)]. From now on, we set k such that $k \equiv \bar{m} - 1 \leq n$, and rewrite [27, Eq. (8.2)] as

$$\begin{aligned} &r_k(\widehat{y}_k) \equiv d_k - B_k\widehat{y}_k, \quad d_k \equiv d + \Delta d_k(\widehat{y}_k), \quad B_k \equiv B\dot{V}_k + \Delta V_k^{(3)}(\widehat{y}_k), \\ &\|\Delta d_k(\widehat{y}_k)\|_2 \leq \widetilde{\gamma}_{kn}^g\|d\|_2, \quad \Delta V_k^{(3)}(y) \equiv \Delta V_k^{(2)} + \Delta C_k(y), \\ &\|\Delta V_k^{(3)}\|_F \lesssim (k^{1/2}\epsilon_p + \widetilde{\gamma}_{kn}^g)\|B\|_F, \end{aligned} \tag{3.7}$$

where $\Delta C_k(y)$ and $\Delta d_k(y)$ are the errors in the MGS least squares solution [27, sec. 7], and where $\widehat{y}_k$ is the computed least squares solution at the kth iteration

$$\widehat{y}_k = \mathrm{argmin}_y \|d - B\dot{V}_k y\|_2, \quad k < \bar{m}.$$

Using the scaling invariance of MGS to scale the right-hand side d_k by some scalar ϕ' and making use of [27, Thm. 2.4] gives a bound on the residual [27, Eq. (8.9)]

$$\|r_k(\hat{y}_k)\|_2^2 \leq (\widetilde{\gamma}_{kn}^g)^2(\|d_k\phi'\|_2^2 + \|B_k\|_F^2)2(\phi')^{-2}. \tag{3.8}$$

In addition to bounding $\|B_k\|_F^2$ and $\|d_k\phi'\|_2^2$, we use the nonsingularity condition (3.3) (which is an upper-bound of the condition number $\|B\|_F\|B^{-1}\|_2$ [9]) in the same fashion as [27, Eq. (8.11)] to compute a bound for $(\phi')^{-2}$, which allows us to rewrite [27, Eq. (8.12)] as

$$\|r_k(\hat{y}_k)\|_2 \lesssim \widetilde{\gamma}_{kn}^g(\|B\|_F\|\hat{y}_k\|_2 + \|d\|_2). \tag{3.9}$$

Since ϵ_p appears in second order terms and higher, they have been dropped, making (3.9) equivalent to [27, Eq. (8.12)]. Considering now $\widehat{x}_k = \mathrm{fl}(\widehat{V}_k\widehat{y}_k) = (\widehat{V}_k + \Delta V_k^{(4)})\widehat{y}_k$ and using a standard matrix–vector product in precision u_g satisfying $\|\Delta V_k^{(4)}\|_F \leq \widetilde{\gamma}_k^g\|\widehat{V}_k\|_F$ and $\Delta B_k \equiv [\Delta V_k^{(3)}(\widehat{y}_k) - B(\Delta V_k^{(4)} + \widehat{V}_k - \dot{V}_k)]\widehat{y}_k\frac{\widehat{x}_k^T}{\|\widehat{x}_k\|_2^2}$, we can rewrite [27, Eq. (8.15)] as

$$\begin{aligned} r_k(\widehat{y}_k) &= d + \Delta d_k(\widehat{y}_k) - (B + \Delta B_k)\widehat{x}_k, \\ \|r_k(\widehat{y}_k)\|_2 &\lesssim \widetilde{\gamma}_{kn}^g(\|B\|_F\|\widehat{x}_k\|_2 + \|d\|_2), \\ \|\Delta d_k(\widehat{y}_k)\|_2 &\leq \widetilde{\gamma}_{kn}^g\|d\|_2, \\ \|\Delta B_k\|_F &\lesssim (k^{1/2}\epsilon_p + \widetilde{\gamma}_{kn}^g)\|B\|_F. \end{aligned} \tag{3.10}$$

This leads to (3.4) and completes the proof of the theorem. ∎

In the original analysis, $\mathrm{fl}(Bv)$ is a standard matrix–vector product operation and $u_p = u_g$, so $\mathrm{fl}(Bv) = (B + \Delta B)v$ holds where $\|\Delta B\|_F \leq \gamma_q^g\|B\|_F$ [14, sec. 3.5]. In this case we can set $f = \Delta Bv$ and apply Theorem 3.1 with $\epsilon_p = \gamma_q^g$, recovering the result of [27] for an unpreconditioned MGS-GMRES in uniform precision, which produces a solution $\widehat{x}_k$ of the system $Bx = d$ satisfying

$$(B + \Delta B)\widehat{x}_k = d + \Delta d, \quad \|\Delta B\|_F \lesssim \widetilde{\gamma}_{kn}^g\|B\|_F, \quad \|\Delta d\|_2 \lesssim \widetilde{\gamma}_{kn}^g\|d\|_2.$$

3.2. Error analysis of GMRES-IR5 with general u_g and u_p precisions. We proceed in three steps. First, we bound the error in $\widehat{s}_i = \mathrm{fl}(\widehat{U}^{-1}\,\mathrm{fl}(\widehat{L}^{-1}\widehat{r}_i))$. Second, we use our analysis of MGS-GMRES of the previous section to prove the backward stability of the solution to the system $\widetilde{A}d_i = \widehat{s}_i$. Third, we combine the previous two results to derive bounds of the type (2.1a)–(2.1b) for the solution of $\widetilde{A}d_i = s_i$.

We begin by bounding the error introduced in forming the preconditioned right-hand side $s_i = \widehat{U}^{-1}\widehat{L}^{-1}\widehat{r}_i$ in precision u_p. The computed $\widehat{s}_i$ satisfies [14, Thm. 8.5]

$$(\widehat{L} + \Delta L)(\widehat{U} + \Delta U)\widehat{s}_i = \widehat{r}_i, \quad |\Delta L| \le \gamma_n^p |\widehat{L}|, \quad |\Delta U| \le \gamma_n^p |\widehat{U}|. \tag{3.11}$$

Also, considering [14, Thm. 9.3] the LU factors computed at precision u_f satisfy

$$\widehat{L}\widehat{U} = A + \Delta A^{(1)}, \quad |\Delta A^{(1)}| \le \gamma_n^f |\widehat{L}||\widehat{U}|. \tag{3.12}$$

Note that, technically, $\widehat{L}$ and $\widehat{U}$ are not the computed factors of A, but the ones of A cast in precision u_f. As the condition number of the cast matrix can be substantially lower due to a regularization effect (see [2]), $\kappa_\infty(A)$ is sometimes an overestimate in the following analysis. We have

$$\begin{aligned} s_i - \widehat{s}_i &= \widehat{U}^{-1}\widehat{L}^{-1}(\Delta L\widehat{U} + \widehat{L}\Delta U + \Delta L\Delta U)\widehat{s}_i \\ &= (A + \Delta A^{(1)})^{-1}(\Delta L\widehat{U} + \widehat{L}\Delta U + \Delta L\Delta U)\widehat{s}_i \\ &\approx (A^{-1} - A^{-1}\Delta A^{(1)}A^{-1})(\Delta L\widehat{U} + \widehat{L}\Delta U + \Delta L\Delta U)\widehat{s}_i \end{aligned}$$

and dropping second-order terms we obtain

$$\begin{aligned} \|s_i - \widehat{s}_i\|_\infty &\lesssim \gamma_{2n}^p \||A^{-1}||\widehat{L}||\widehat{U}|\|_\infty \|\widehat{s}_i\|_\infty \\ &\le n^2\rho_n\widetilde{\gamma}_n^p\kappa_\infty(A)\|\widehat{s}_i\|_\infty \lesssim n^2\rho_n\widetilde{\gamma}_n^p\kappa_\infty(A)\|s_i\|_\infty. \end{aligned} \tag{3.13}$$

where the second inequality comes from [14, Lem. 9.6]. Actually, in the previous bound, certain terms of order $\kappa(A)^2 u_p u_f$ are dropped (as second order terms). While our assumptions allow $\kappa(A)u_f$ to be arbitrarily large and these terms to not be necessarily negligible, we observe in practice that $\kappa(A)u_f$ is generally at most of order a constant in these terms. This appears to be the result of the regularization effect mentioned earlier.

Next, we show that this new version of GMRES (with general u_p and u_g precisions) provides a backward stable solution to the system $\widetilde{A}d_i = s_i$, where $\widetilde{A} = \widehat{U}^{-1}\widehat{L}^{-1}A$ and $s_i = \widehat{U}^{-1}\widehat{L}^{-1}r_i$. We rely on Theorem 3.1, which provides backward error bounds for MGS-GMRES with general matrix–vector products. Our aim is therefore to prove that (3.2) holds for some ϵ_p when the matrix–vector products are computed with matrix $\widetilde{A} = \widehat{U}^{-1}\widehat{L}^{-1}A$ and in precision u_p. Let $z_j = \widetilde{A}\widehat{v}_j$ be computed in precision u_p by a matrix product followed by two triangular solves. Then

$$\begin{aligned} (A + \Delta A^{(2)})\widehat{v}_j &= \widehat{w}_j, \quad |\Delta A^{(2)}| \le \gamma_q^p |A|, \\ (\widehat{L} + \Delta L)\widehat{y}_j &= \widehat{w}_j, \quad |\Delta L| \le \gamma_n^p |\widehat{L}|, \\ (\widehat{U} + \Delta U)\widehat{z}_j &= \widehat{y}_j, \quad |\Delta U| \le \gamma_n^p |\widehat{U}|. \end{aligned}$$

The computed vector $\widehat{z}_j$ can therefore be written as

$$\begin{aligned}\widehat{z}_j &= (\widehat{U}+\Delta U)^{-1}(\widehat{L}+\Delta L)^{-1}(A+\Delta A^{(2)})\widehat{v}_j \\ &\approx (\widehat{U}^{-1}-\widehat{U}^{-1}\Delta U\widehat{U}^{-1})(\widehat{L}^{-1}-\widehat{L}^{-1}\Delta L\widehat{L}^{-1})(A+\Delta A^{(2)})\widehat{v}_j \\ &= \widetilde{A}\widehat{v}_j + f_j,\end{aligned}$$

where

$$\begin{aligned}f_j &\approx \big(\widehat{U}^{-1}\widehat{L}^{-1}\Delta A^{(2)} - \widehat{U}^{-1}\widehat{L}^{-1}\Delta L\widehat{L}^{-1}A - \widehat{U}^{-1}\Delta U\widehat{U}^{-1}\widehat{L}^{-1}A\big)\widehat{v}_j \\ &= \big(\widetilde{A}A^{-1}\Delta A^{(2)} - \widehat{U}^{-1}\widehat{L}^{-1}\Delta L\widehat{U}\widetilde{A} - \widehat{U}^{-1}\Delta U\widetilde{A}\big)\widehat{v}_j,\end{aligned}$$

and so

$$\|f_j\|_2 \lesssim \gamma_n^p(\kappa_F(A)+\kappa_F(\widehat{U})\kappa_F(\widehat{L})+\kappa_F(\widehat{U}))\|\widetilde{A}\|_F\|\widehat{v}_j\|_2.$$

Since we can expect $\kappa_F(\widehat{L})$ to be of modest size (because $\widehat{L}$ is a unit triangular matrix with off-diagonal elements bounded by 1 if, for example, partial pivoting is used in the factorization), and since $\kappa_F(\widehat{U}) \lesssim \kappa_F(A)\kappa_F(\widehat{L})$, we obtain

$$\|f_j\|_2 \lesssim \widetilde{\gamma}_n^p\kappa_F(A)\|\widetilde{A}\|_F\|\widehat{v}_j\|_2 \le n\widetilde{\gamma}_n^p\kappa_\infty(A)\|\widetilde{A}\|_F\|\widehat{v}_j\|_2. \tag{3.14}$$

Condition (3.2) is thus satisfied for $\epsilon_p = n\widetilde{\gamma}_n^p\kappa_\infty(A)$, and Theorem 3.1 is therefore applicable. From (3.4) we obtain

$$(\widetilde{A}+\Delta\widetilde{A})\widehat{d}_i = \widehat{s}_i + \Delta\widehat{s}_i, \tag{3.15a}$$

$$\|\Delta\widetilde{A}\|_F \lesssim (\widetilde{\gamma}_{kn}^g + n\widetilde{\gamma}_{k^{1/2}n}^p\kappa_\infty(A))\|\widetilde{A}\|_F, \tag{3.15b}$$

$$\|\Delta\widehat{s}_i\|_2 \lesssim \widetilde{\gamma}_{kn}^g\|\widehat{s}_i\|_2 \lesssim n^{1/2}\widetilde{\gamma}_{kn}^g\|s_i\|_\infty. \tag{3.15c}$$

Rewriting (3.15a) as

$$s_i - \widetilde{A}\widehat{d}_i = \Delta\widetilde{A}\widehat{d}_i - (\widehat{s}_i - s_i) - \Delta\widehat{s}_i \tag{3.16}$$

and using (3.15b), (3.13), and (3.15c) to bound the three terms on the right-hand side, we obtain

$$\begin{aligned}\|s_i - \widetilde{A}\widehat{d}_i\|_\infty &\le \|\Delta\widetilde{A}\|_\infty\|\widehat{d}_i\|_\infty + \|\widehat{s}_i - s_i\|_\infty + \|\Delta\widehat{s}_i\|_\infty \\ &\lesssim n\big(\widetilde{\gamma}_{kn}^g + n\widetilde{\gamma}_{k^{1/2}n}^p\kappa_\infty(A)\big)\|\widetilde{A}\|_\infty\|\widehat{d}_i\|_\infty \\ &\quad + n^2\rho_n\widetilde{\gamma}_n^p\kappa_\infty(A)\|s_i\|_\infty + n^{1/2}\widetilde{\gamma}_{kn}^g\|s_i\|_\infty \\ &\lesssim n^3\max(k^{1/2},\rho_n)\big(u_g + u_p\kappa_\infty(A)\big)\big(\|\widetilde{A}\|_\infty\|\widehat{d}_i\|_\infty + \|s_i\|_\infty\big).\end{aligned}$$

In conclusion, the normwise relative backward error of the system $\widetilde{A}\widehat{d}_i = s_i$ is bounded by

$$\frac{\|s_i - \widetilde{A}\widehat{d}_i\|_\infty}{\|\widetilde{A}\|_\infty\|\widehat{d}_i\|_\infty + \|s_i\|_\infty} \lesssim f(n,k,\rho_n)\big(u_g + u_p\kappa_\infty(A)\big) \tag{3.17}$$

and the relative error of the computed $\widehat{d}_i$ therefore satisfies

$$\frac{\|d_i - \widehat{d}_i\|_\infty}{\|d_i\|_\infty} \lesssim f(n,k,\rho_n)\big(u_g + u_p\kappa_\infty(A)\big)\kappa_\infty(\widetilde{A}), \tag{3.18}$$

where $f(n,k,\rho_n) = n^3\max(k^{1/2},\rho_n)$.

3.3. Convergence conditions on $\kappa(A)$. We can now use this analysis together with (2.2) and (2.3) to determine sufficient conditions for the convergence of the forward and backward errors for general u_p and u_g parameters.

Beginning with the forward error, (3.18) shows that (2.1a) holds with

$$u_s\|E_i\|_\infty \equiv f(n,k,\rho_n)\big(u_g + u_p\kappa_\infty(A)\big)\kappa_\infty(\widetilde{A}),$$

and so, dropping constants, (2.2) becomes

$$(u_g + u_p\kappa(A))\kappa(\widetilde{A}) \lll 1. \tag{3.19}$$

By using the bound (2.5) on $\kappa(\widetilde{A})$, we obtain the condition

$$(u_g + u_p\kappa(A))(1+\kappa(A)^2u_f^2) \lll 1 \tag{3.20}$$

for the forward error to converge to its limiting value of order $qu_r\,\mathrm{cond}(A,x)+u$.

Next we determine a condition for the backward error to converge. First we need to bound the backward error of the original correction equation $Ad_i = r_i$. By observing that the computed residual $\widehat{r}_i$ satisfies $\widehat{r}_i - A\widehat{d}_i = \widehat{L}\widehat{U}(s_i - \widetilde{A}\widehat{d}_i)$, since $s_i = \widehat{U}^{-1}\widehat{L}^{-1}\widehat{r}_i$, and using the bound (3.17) and the relation (3.12) we obtain

$$\begin{aligned}\|\widehat{r}_i - A\widehat{d}_i\|_\infty &\lesssim f(n,k,\rho_n)\big(u_g + u_p\kappa_\infty(A)\big)\|\widehat{L}\widehat{U}\|_\infty(\|\widetilde{A}\|_\infty\|\widehat{d}_i\|_\infty + \|s_i\|_\infty)\\ &\lesssim f(n,k,\rho_n)\big(u_g + u_p\kappa_\infty(A)\big)(\|\widetilde{A}\|_\infty\|A\|_\infty\|\widehat{d}_i\|_\infty + \kappa_\infty(A)\|\widehat{r}_i\|_\infty).\end{aligned}$$

We have thus shown that (2.1b) holds with $u_s = u_g + u_p\kappa_\infty(A)$, $c_1 = f(n,k,\rho_n)\|\widetilde{A}\|_\infty$, and $c_2 = f(n,k,\rho_n)\kappa_\infty(A)$. Dropping constants, condition (2.3) then becomes

$$\big(u_g + u_p\kappa(A)\big)(\|\widetilde{A}\| + 1)\kappa(A) \lll 1.$$

Using (3.12) we have

$$\widetilde{A} = \widehat{U}^{-1}\widehat{L}^{-1}A = (A + \Delta A^{(1)})^{-1}A \approx I - A^{-1}\Delta A^{(1)}$$

and using [14, Lem. 9.6] we obtain the bound

$$\|\widetilde{A}\|_\infty \lesssim 1 + \|A^{-1}\|_\infty\|\Delta A^{(1)}\|_\infty \le 1 + f(n,\rho_n)u_f\kappa_\infty(A).$$

We finally obtain the condition

$$\big(u_g + u_p\kappa(A)\big)\big(1+\kappa(A)u_f\big)\kappa(A) \lll 1 \tag{3.21}$$

for the backward error to converge to its limiting value of order $qu_r + u$.

We summarize the error analysis of this section in the following theorem.

THEOREM 3.2. *Let $Ax = b$ be solved by GMRES-IR5 (Algorithm 3.1) using MGS-GMRES at step 5. Provided that $u_g \ge u$, if*

$$\phi_1 \equiv (u_g + u_p\kappa(A))(1+\kappa(A)^2u_f^2) \lll 1, \tag{3.22}$$

the forward error will be reduced at each iteration by a factor approximately ϕ_1 until it reaches its limiting value $qu_r\,\mathrm{cond}(A,x)+u$, and if

$$\phi_2 \equiv \big(u_g + u_p\kappa(A)\big)\big(1+\kappa(A)u_f\big)\kappa(A) \lll 1, \tag{3.23}$$

the backward error will be reduced at each iteration by a factor approximately ϕ_2 until it reaches its limiting value $qu_r + u$.

TABLE 3.1

Convergence conditions and limiting backward and forward errors for the LU-IR3, GMRES-IR5, and GMRES-IR3 algorithms. The convergence conditions of the first and third algorithms are retrieved from the analyses of [4], [5] that we summarized in section 2.

	Backward error	Forward error
LU-IR3	$u_f\kappa(A) \lll 1$	$u_f\kappa(A) \lll 1$
GMRES-IR5	$\big(u_g + u_p\kappa(A)\big)\big(1+\kappa(A)u_f\big)\kappa(A) \lll 1$	$(u_g + u_p\kappa(A))(1+\kappa(A)^2u_f^2) \lll 1$
GMRES-IR3	$u^{1/2}u_f^{1/2}\kappa(A) \lll 1$	$u^{1/2}u_f\kappa(A) \lll 1$
Limiting values	$qu_r + u$	$qu_r\mathrm{cond}(A,x) + u$

Curiously, condition (3.23) is stricter than the condition for the forward error, since (3.22) has an extra u_f term. However, note that since the backward error is bounded by the forward error (see, for example, [14, Prob. 7.2]), (3.22) is also an obvious condition for the backward error to converge to the limiting value of the forward error, of order $q\,\mathrm{cond}(A,x)u_r + u$. In particular, if $u_r = u^2$, condition (3.23) is not useful because (3.22) also guarantees a backward error of order u with a less restrictive condition on $\kappa(A)$, since by assumption $\kappa(A)u < 1$.

As a check, we compare our results with [16], which analyzes the case $u_p = u_g = u_r = u$, that is, GMRES-IR in two precisions, and uses a different argument to that here. Considering $1 \lll \kappa(A)u_f$, our condition (3.22) for convergence of the forward error is $\kappa(A)^3uu_f^2 \lll 1$ and our condition (3.23) for convergence of the backward error is $\kappa(A)^3uu_f \lll 1$. These conditions agree with [16, Eqs (3.8), (3.6)].

We also compare with the analysis of [5] for GMRES-IR3, the conclusions from which we summarized in section 2. For $u_g = u$ and $u_p = u^2$, our condition (3.22) for convergence of the forward error is $(u+\kappa(A)u^2)(1+\kappa(A)^2u_f^2) \lll 1$, and requires in particular $\kappa(A)^2uu_f^2 \lll 1$, since we are assuming $\kappa(A)u < 1$. This agrees with (2.6). Our condition (3.23) for convergence of the backward error requires, similarly, $\kappa(A)^2uu_f \lll 1$, which agrees with condition (2.7).

Finally, we summarize in Table 3.1 convergence conditions and limiting backward and forward errors for LU-IR3, GMRES-IR3, and GMRES-IR5. Since the limiting backward error and forward error are independent of the correction equation solver (as long as the conditions (2.1) are satisfied), they are the same for the three algorithms. As explained in the previous paragraph, with $u_g = u$ and $u_p = u^2$, the convergence conditions of GMRES-IR5 reduce to those of GMRES-IR3. In addition, as long as $u_p\kappa(A) \lll 1$, the forward error convergence condition is always better than the one of LU-IR3 (we need more assumptions on u_g and u_p to have the same conclusion for the backward error). Hence, the convergence conditions of GMRES-IR5 achieve a tradeoff between those of LU-IR3 and those of GMRES-IR3. A thorough comparison of these convergence conditions is the major concern of the rest of this article.

3.4. Identifying meaningful combinations of precisions. Table 3.2 shows the most widely used floating-point arithmetics; most modern supercomputers provide hardware support for at least three of them. Assuming all these arithmetics can be used, GMRES-IR5 has thousands of different combinations of its five precision parameters. Among these, not all are relevant and, therefore, it is important to identify the subset of meaningful combinations. We say a combination is meaningful if none of the precisions it employs can be reduced without degrading the numerical properties (convergence condition or limiting accuracy). Every meaningful combination attains a tradeoff between performance, robustness (ability to converge for ill-conditioned

TABLE 3.2
Parameters for floating-point arithmetics: symbol used in this paper, number of bits in significand, number of bits in exponent, unit roundoff, and range.

Arithmetic	Symbol	Significand	Exponent	Unit roundoff	Range
bfloat16	B	8	8	3.91×10^{-3}	$10^{\pm 38}$
fp16	H	11	5	4.88×10^{-4}	$10^{\pm 5}$
fp32	S	24	8	5.96×10^{-8}	$10^{\pm 38}$
fp64	D	53	11	1.11×10^{-16}	$10^{\pm 308}$
fp128	Q	113	15	9.63×10^{-35}	$10^{\pm 4932}$

matrices), and accuracy (ability to converge to small errors).

As an example, the meaningful combinations for LU-IR3 must satisfy $u^2 \leq u_r \leq u \leq u_f$. Indeed, the limiting backward and forward errors (recalled in Table 3.1) show that we should have $u_r \leq u$ and that setting $u_r < u^2$ is not useful since $u_r = u^2$ is already enough to ensure a forward error of order u (since by assumption $\kappa(A)u < 1$).

Meaningful combinations for GMRES-IR5 also satisfy $u^2 \leq u_r \leq u \leq u_f$, so our aim now is to discuss the choice of the two new precision parameters $u_g \geq u$ and $u_p \geq u^2$. To compute the bounds on $\kappa(A)$ given by the conditions (3.22) and (3.23), we solve the equalities $(u_g + u_p x)(1 + u_f^2 x^2) = 1$ and $(u_g + u_p x)(1 + u_f x)x = 1$, respectively. We now state some observations that can be deduced from our analysis, in particular from condition (3.22).

- $u_p \leq u_g$. This first observation comes from the term $u_g + u_p\kappa(A)$ that appears in the convergence conditions. We would like the components of this term to be balanced, so that $u_p \approx u_g/\kappa(A) \leq u_g$. So $u_p < u_g$ may be required, but $u_p > u_g$ is not meaningful. We also note that there is no advantage to taking $u_p < u_g/\kappa(A)$.
- $u_p < u_f$. This second observation comes from the fact that if $u_p = u_f$, condition (3.19) requires $\kappa(A)u_f\kappa(\widetilde{A}) \lll 1$, which is worse than the condition $\kappa(A)u_f \lll 1$ for LU-IR3.
- $u_p < u$, $u_p = u$, and $u_p > u$ are all meaningful. This is one of the main conclusions of our analysis. We know from [5] (in which $u_p = u^2$ and $u_g = u$) that setting $u_p = u^2$ provides the least restrictive convergence conditions (2.6) and (2.7), but the precise role of u_p in the convergence was not analyzed. With the new conditions (3.22) and (3.23) obtained from our generalized analysis, we can now understand what the conditions become if u_p is taken larger than u^2. Crucially, setting $u_p = u$ and even $u_p > u$ can potentially yield conditions that remain less restrictive than the LU-IR3 condition $\kappa(A)u_f \lll 1$, and therefore represent meaningful combinations. Let us illustrate this observation with a practical example. Assume u_f is set to fp16 and $u = u_g$, with u_g set to fp64. Then the condition for the forward error to converge with LU-IR3 is $\kappa(A) \lll 2 \times 10^3$. Instead, with GMRES-IR5:
  - If u_p is set to fp128 (as in [5]), condition (3.22) becomes $\kappa(A) \lll 2\times 10^{11}$. We recover the same condition as in [5].
  - If u_p is set to fp64 (and thus $u_p = u$), condition (3.22) becomes $\kappa(A) \lll 3 \times 10^7$, which is still much better than the LU-IR3 condition. This version of GMRES-IR was analyzed in [16] and has been successfully used in practice, for example in [11], [12].
  - If u_p is set to fp32 (and thus $u_p > u$), condition (3.22) becomes $\kappa(A) \lll 4\times 10^4$, which is still over an order of magnitude better than the LU-IR3

condition. Note that this variant uses up to four different precisions if u_r is set to fp128. Note also that this combination is actually not meaningful with u_g set to fp64 (because $u_p > u_g$), but it becomes meaningful by switching u_g to fp32 (and thus $u_g = u_p > u$), see the next bullet.

These examples illustrate how the u_p precision can be tuned to achieve different levels of tradeoff between robustness (ability to handle ill-conditioned matrices) and performance (cost of the application of LU factors within GMRES).

- $u_g = u$ and $u_g > u$ are both meaningful. This is also an important conclusion of our analysis. Whereas in [5], u_g is set to u to obtain the least restrictive convergence condition, our analysis reveals that setting $u_g > u$ can also be meaningful. In fact, as long as $u_g < 1$, condition (3.22) is better than the LU-IR3 condition $\kappa(A)u_f \lll 1$, so we have much flexibility in choosing u_g. Let us again illustrate this with a practical example. Assume u_f is set to fp16 and $u = u_p$ are set to fp64. With $u_g = u$, as previously stated, the condition (3.22) is $\kappa(A) \lll 3 \times 10^7$. However, if u_g is set to fp32, the condition is $\kappa(A) \lll 8 \times 10^6$, which is only slightly worse and remains much better than the LU-IR3 condition. The precision u_g can be chosen even lower, for example if u_g is set to fp16 (i.e., $u_g = u_f$), the condition becomes $\kappa(A) \lll 9 \times 10^4$ and still remains better than the LU-IR3 one. Note that with u_g set to fp32, setting u_p to fp128 instead of fp64 does not improve the condition $\kappa(A) \lll 8 \times 10^6$: this shows that the meaningful values of u_p can be influenced by the choice of u_g, and vice versa. In conclusion the u_g precision can also be tuned to achieve different levels of tradeoff between robustness and performance (the cost of GMRES and, in particular, the memory footprint of the Krylov basis).
- $u_g < u_f$, $u_g = u_f$, and $u_g > u_f$ are all meaningful. This final observation is that u_g and u_f can be independent. We have already illustrated in previous examples where u_f is fp16 that $u_g < u_f$ and $u_g = u_f$ are both meaningful. This final observation states that even $u_g > u_f$ can be meaningful. To see why, let us take another example with u_f set to fp32 and $u = u_p$ set to fp64. Then setting u_g to fp16 yields the condition $\kappa(A) \lll 7 \times 10^8$, which is better than any combination with the same u and u_p but with u_f set to fp16 (the best possible condition being $\kappa(A) \lll 3 \times 10^7$ for u_g at least in fp64). Usually, one would expect an fp16 LU factorization and fp32 GMRES to be faster than the converse (fp32 factorization and fp16 GMRES), so in practice the combinations with $u_g > u_f$ are only relevant for a narrow range of $\kappa(A)$ (in this example, for $\kappa(A)$ such that $\kappa(A) \lll 3 \times 10^7$ is not satisfied but $\kappa(A) \lll 7 \times 10^8$ is).

In Table 3.3, we summarize all the possible combinations of u_p and u_g when u_f is set to bfloat16, fp16, or fp32 (these three precisions being the ones that determine the convergence conditions). The underlined, red terms correspond to combinations of these three precisions that are not meaningful. Interestingly, when the working precision u is fp64 and the residual precision u_r is fp128, Table 3.3 contains thirteen meaningful variants that use at least four different arithmetics, including two that use all five (BHS and HBS).

All the observations made in this part reduce the number of meaningful combinations of the five precision parameters to a subset of 112 combinations, corresponding to 3.5% of all the possibilities. We summarize the numerical properties (convergence condition and limiting accuracy, for both the forward and backward errors) of a se-

TABLE 3.3

Bound on $\kappa(A)$, rounded to one significant figure, given by conditions (3.22) and (3.23) for the forward and backward errors to converge with GMRES-IR5, depending on the precisions u_f, u_g, and u_p, and assuming the working precision u is double. We recall that the forward error convergence condition for LU-IR3 is $\kappa(A) \lll 3 \times 10^2$ for $u_f =$ B, $\kappa(A) \lll 2 \times 10^3$ for $u_f =$ H, and $\kappa(A) \lll 2 \times 10^7$ for $u_f =$ S. The red, underlined terms denote combinations of precisions that are not meaningful. Each variant is presented in the form of a triplet (u_f, u_g, u_p), hence HDQ *means $u_f =$ H, $u_g =$ D, and $u_p =$ Q. We recall that u_r does not play a role in the convergence bounds.*

	Variants (u_f, u_g, u_p)	Forward	Backward	
$u_f =$ B	BBH BHH BSH BDH	5×10^2	4×10^1	§
	BBS BBD BBQ	4×10^3	2×10^2	
	BHS	8×10^3	6×10^2	
	BHD BHQ	1×10^4	6×10^2	
	BSS BDS	1×10^4	2×10^3	
	BSD BSQ	1×10^6	7×10^4	
	BDD	8×10^6	1×10^6	
	BDQ	2×10^{10}	2×10^9	
$u_f =$ H	HBS HBD HBQ	3×10^4	2×10^2	
	HHS	4×10^4	1×10^3	
	HSS HDS	4×10^4	3×10^3	
	HHD HHQ	9×10^4	1×10^3	
	HSD HSQ	8×10^6	2×10^5	
	HDD	3×10^7	3×10^6	
	HDQ	2×10^{11}	4×10^9	
$u_f =$ S	SBD SBQ	3×10^8	3×10^2	
	SHD SHQ	7×10^8	2×10^3	
	SSD	1×10^{10}	1×10^7	
	SDD	1×10^{10}	5×10^7	
	SSQ	7×10^{10}	1×10^7	
	SDQ	2×10^{15}	4×10^{11}	

§ BHH is slightly better than BBH on Forward and Backward before rounding.

lected subset of 42 GMRES-IR5 variants in Table 3.4. Importantly, the table includes variants used in existing implementations, for which our analysis provides new theoretical guarantees. It also includes new combinations of precisions not proposed previously that achieve new, finer tradeoffs between the convergence conditions and the precisions used.

In order to select the appropriate combination of precisions, we have shown in this section that the user can choose within the subset of the meaningful combinations those providing the appropriate convergence condition and forward and backward error bounds according to the application requirements. Within this selection the user can take into account hardware and software features to further refine the selection.

4. Numerical experiments. We now perform numerical experiments to assess the validity of the convergence conditions of GMRES-IR5 derived in the previous section. Throughout our experiments, we focus on the forward error convergence (condition (3.22)) and we fix $u =$ D and $u_r =$ Q and analyze the role of the factorization precision u_f and that of the newly introduced precisions u_p (preconditioned matrix–vector product precision) and u_g (GMRES precision).

We have written a Julia code that implements GMRES-IR5 and LU-IR3, where half precision arithmetics (fp16 and blfoat16) are simulated. We have made this code publicly available[1].

[1] https://github.com/bvieuble/Itref.jl

TABLE 3.4

Bounds on $\kappa(A)$, rounded to one significant figure, such that GMRES-IR5 with the precisions in the first five columns is guaranteed to converge to the indicated limiting error values.

					Forward error		Backward error	
u	u_r	u_f	u_g	u_p	$\kappa(A)$	Limit	$\kappa(A)$	Limit
S	S	B	B	S	4×10^3	$\text{cond}(A,x) \times$ S	2×10^2	S
S	S	B	H	S	8×10^3	$\text{cond}(A,x) \times$ S	6×10^2	S
S	S	B	S	S	1×10^4	$\text{cond}(A,x) \times$ S	2×10^3	S
S	S	H	H	S	4×10^4	$\text{cond}(A,x) \times$ S	1×10^3	S
S	S	H	S	S	4×10^4	$\text{cond}(A,x) \times$ S	3×10^3	S
S	S	B	S	D	1×10^6	$\text{cond}(A,x) \times$ S	7×10^4	S
S	S	H	S	D	8×10^6	$\text{cond}(A,x) \times$ S	2×10^5	S
S	D	B	B	S	4×10^3	S	2×10^2	S
S	D	B	H	S	8×10^3	S	6×10^2	S
S	D	B	S	S	1×10^4	S	2×10^3	S
S	D	H	H	S	4×10^4	S	1×10^3	S
S	D	H	S	S	4×10^4	S	3×10^3	S
S	D	B	S	D	1×10^6	S	7×10^4	S
S	D	H	S	D	8×10^6	S	2×10^5	S
D	D	B	B	S	4×10^3	$\text{cond}(A,x) \times$ D	2×10^2	D
D	D	B	H	S	8×10^3	$\text{cond}(A,x) \times$ D	6×10^2	D
D	D	B	S	S	1×10^4	$\text{cond}(A,x) \times$ D	2×10^3	D
D	D	H	H	S	4×10^4	$\text{cond}(A,x) \times$ D	1×10^3	D
D	D	H	S	S	4×10^4	$\text{cond}(A,x) \times$ D	3×10^3	D
D	D	B	S	D	1×10^6	$\text{cond}(A,x) \times$ D	7×10^4	D
D	D	H	S	D	8×10^6	$\text{cond}(A,x) \times$ D	2×10^5	D
D	D	B	D	D	8×10^6	$\text{cond}(A,x) \times$ D	1×10^6	D
D	D	H	D	D	3×10^7	$\text{cond}(A,x) \times$ D	3×10^6	D
D	D	S	H	D	7×10^8	$\text{cond}(A,x) \times$ D	2×10^3	D
D	D	S	D	D	1×10^{10}	$\text{cond}(A,x) \times$ D	5×10^7	D
D	D	B	D	Q	2×10^{10}	$\text{cond}(A,x) \times$ D	2×10^9	D
D	D	H	D	Q	2×10^{11}	$\text{cond}(A,x) \times$ D	4×10^9	D
D	D	S	D	Q	2×10^{15}	$\text{cond}(A,x) \times$ D	4×10^{11}	D
D	Q	B	B	S	4×10^3	D	2×10^2	D
D	Q	B	H	S	8×10^3	D	6×10^2	D
D	Q	B	S	S	1×10^4	D	2×10^3	D
D	Q	H	H	S	4×10^4	D	1×10^3	D
D	Q	H	S	S	4×10^4	D	3×10^3	D
D	Q	B	S	D	1×10^6	D	7×10^4	D
D	Q	B	D	D	8×10^6	D	1×10^6	D
D	Q	H	S	D	8×10^6	D	2×10^5	D
D	Q	H	D	D	3×10^7	D	3×10^6	D
D	Q	S	H	D	7×10^8	D	2×10^3	D
D	Q	S	D	D	1×10^{10}	D	5×10^7	D
D	Q	B	D	Q	2×10^{10}	D	2×10^9	D
D	Q	H	D	Q	2×10^{11}	D	4×10^9	D
D	Q	S	D	Q	2×10^{15}	D	4×10^{11}	D

4.1. Random dense matrices. We first use random dense matrices with specified condition 2-norm number $\kappa(A)$, which are generated in Julia with the command `matrixdepot('randsvd', n, kappa, mode)` where `mode = 2`. Note that this class of matrices leads to unusually large growth factors ρ_n of order n [13]. However, we use only small matrices ($n = 50$), for which ρ_n does not exceed 20.

We take $\kappa(A) = 10^c$, for $c = 0\colon 17$ and, for each value of $\kappa(A)$, we generate 100 random 50×50 matrices of corresponding condition number. Then, we run LU-

IR3 and GMRES-IR5 on each matrix and compute their success rate, that is, the percentage of matrices for which a forward error $\|x - \hat{x}\|_2/\|x\|_2 \leq 4.44 \times 10^{-16}$ is achieved, since with $u = \text{D}$ and $u_r = \text{Q}$ the forward error should reach full double precision with no dependency on $\text{cond}(A, x)$.

Figure 4.1 reports the success rate of LU-IR3 and nine variants of GMRES-IR5, corresponding to all possible combinations of the u_p and u_g parameters over the values $u_p = \text{S,D,Q}$ and $u_g = \text{B,S,D}$, with $u_f = \text{B}$ fixed for all variants. This experiment allows us to obtain an empirical bound on the value of $\kappa(A)$ at which each variant stops being able to converge. For example, the success rate of LU-IR3 is 100% as long as $\kappa(A) \leq 10^2$, but starts decreasing for larger $\kappa(A)$, and quickly becomes 0%. This experimentally confirms the theoretical condition $\kappa(A) \lll 2 \times 10^3$ given by (2.4). Some combinations in Figure 4.1 are not meaningful as defined by section 3.4; we use red underlining in the legend to distinguish them.

Let us now analyze the GMRES-IR5 variants, starting with the role of the preconditioned matrix–vector product precision u_p with fixed $u_g = D$ (top graph of Figure 4.1). We can observe that convergence is achieved with 100% success rate as long as $\kappa(A)$ is smaller than 10^7 for $u_p = \text{S}$ and 10^{15} for $u_p = \text{D}$ or Q. The relative robustness of each method is therefore consistent with the theoretical bounds of Table 3.3. However, the variants with $u_p = \text{S}$ and $u_p = \text{D}$ both perform much better than expected. This is not entirely surprising since the analysis can be pessimistic, especially in the bound (2.5) on $\kappa(\widehat{A})$, which in practice has been observed to often be of order $\kappa(A)u_f$ rather than the worst-case bound $(\kappa(A)u_f)^2$ [4], [26], [28].

Next we analyze the role of the GMRES precision u_g by comparing the top graph of Figure 4.1 with the middle and bottom ones. When $u_p = \text{S}$, switching from $u_g = \text{D}$ (top) to $u_g = \text{S}$ (middle) has no impact on the success rate, which equals 100% as long as $\kappa(A) \leq 10^7$; reducing the precision even further by setting $u_g = \text{B}$ (bottom) only has a slight impact: the success rate remains at 100% for $\kappa(A) \leq 10^5$. When $u_p = \text{D}$ or Q, reducing the GMRES precision has a much more visible impact: a 100% success rate is achieved only when $\kappa(A) \leq 10^9$ ($u_g = \text{S}$, middle) or $\kappa(A) \leq 10^5$ ($u_g = \text{B}$, bottom).

These experiments also show that the success rate is independent of u_p when $u_g = B$, and almost independent of u_g when $u_p = S$. This illustrates that the meaningful choices of u_p depend on the choice of u_g, and vice versa. Overall, the experiments of Figure 4.1 are therefore in good agreement with the theoretical bounds. Importantly, they confirm that even with relaxed requirements on the precisions u_p and u_g, GMRES-IR5 can still handle matrices that are much more ill-conditioned than LU-IR3.

Finally, we evaluate in Figure 4.2 the two meaningful variants of Table 3.3 that use five different precisions: BHS ($u_f = \text{B}$, $u_g = \text{H}$, $u_p = \text{S}$) and HBS ($u_f = \text{H}$, $u_g = \text{B}$, $u_p = \text{S}$). We compare these two variants with LU-IR3 with $u_f = \text{B}$ or H, and with the BBS and HHS GMRES-IR5 variants ($u_f = \text{B}$, $u_g = \text{B}$, $u_p = \text{S}$ and $u_f = \text{H}$, $u_g = \text{H}$, $u_p = \text{S}$), which are the four-precision variants right below and above in terms of convergence condition. The figure experimentally confirms that both of these five-precision variants are in between BBS and HHS and are able to handle more ill-conditioned matrices than LU-IR3, showing that they are meaningful not only theoretically but also in practice.

4.2. Real-life matrices from SuiteSparse. It is also important to check whether reducing the preconditioner and/or GMRES precisions impacts the number of iterations required to converge. To answer this question, we use a set of 230

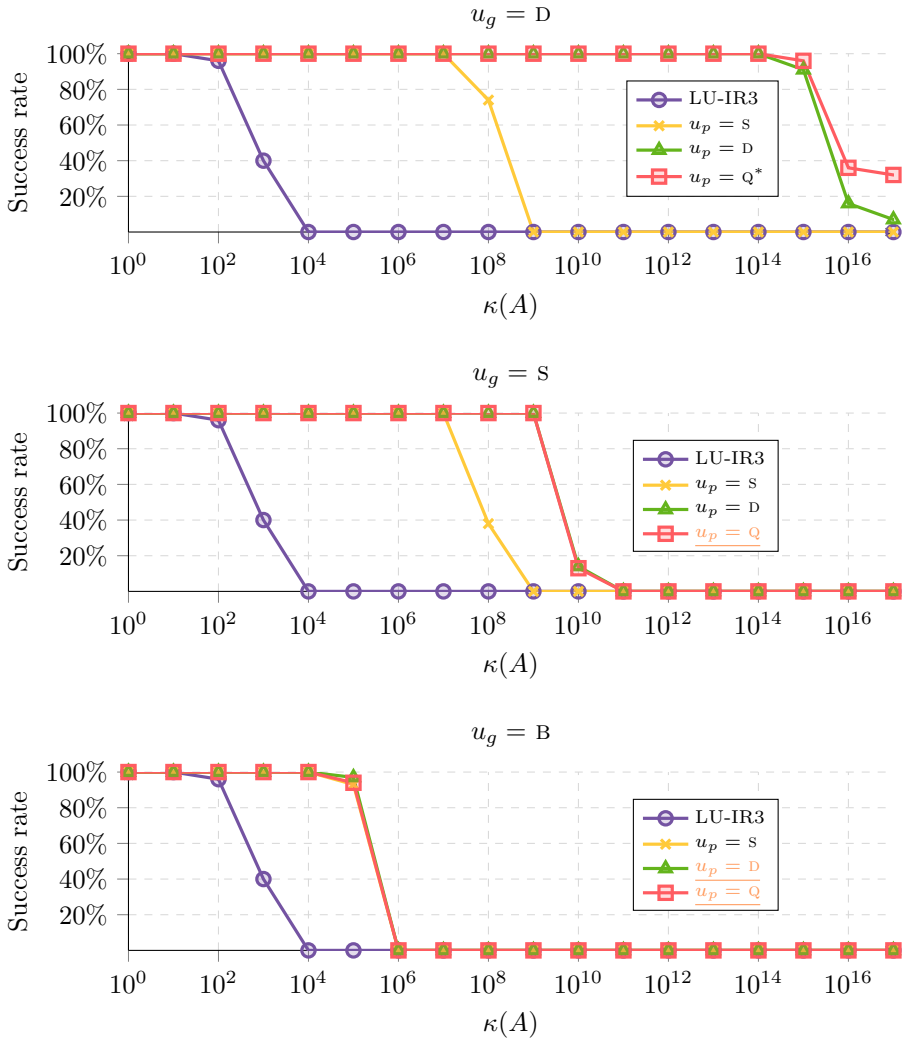


FIG. 4.1. *Proportion of matrices for which LU-IR3 and GMRES-IR5 with varying u_p and u_g converged to double precision forward error as a function of $\kappa(A)$. For all variants, u_f = B, u = D, and u_r = Q. The red, underlined terms denote combinations of precisions that are not meaningful in Table 3.3. The GMRES-IR5 variant with an asterisk "*" is equivalent to GMRES-IR3.*

matrices from the SuiteSparse collection. These matrices are all real, square, and of dimension between 500 and 3000. We compare, over the values u_f = B, H, S, LU-IR3 with a subset of meaningful GMRES-IR5 variants that are selected to illustrate remarkable behaviors and provide insight on the properties of the method. Rather than counting the number of iterations, we count the number of calls to LU triangular solves: this is because in the case of GMRES-IR5, an extra LU solve is required at the start of each IR step (external iterations).

For the experiments that use an LU factorization in fp16 arithmetic, we must pay attention to the narrow range of this arithmetic (see Table 3.2). Many matrices in our set have entries outside of this range, and so it is essential to address this issue. We

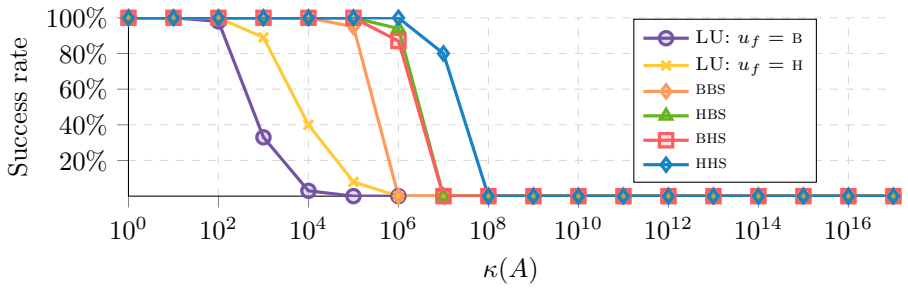


FIG. 4.2. *Proportion of matrices for which LU-IR and GMRES-IR variants (including five meaningful precisions combinations, with u_f, u_g, u_p as specified in the legend) converged to double precision forward error as a function of $\kappa(A)$. For all variants u = D and u_r = Q and all the variants in this figure are meaningful.*

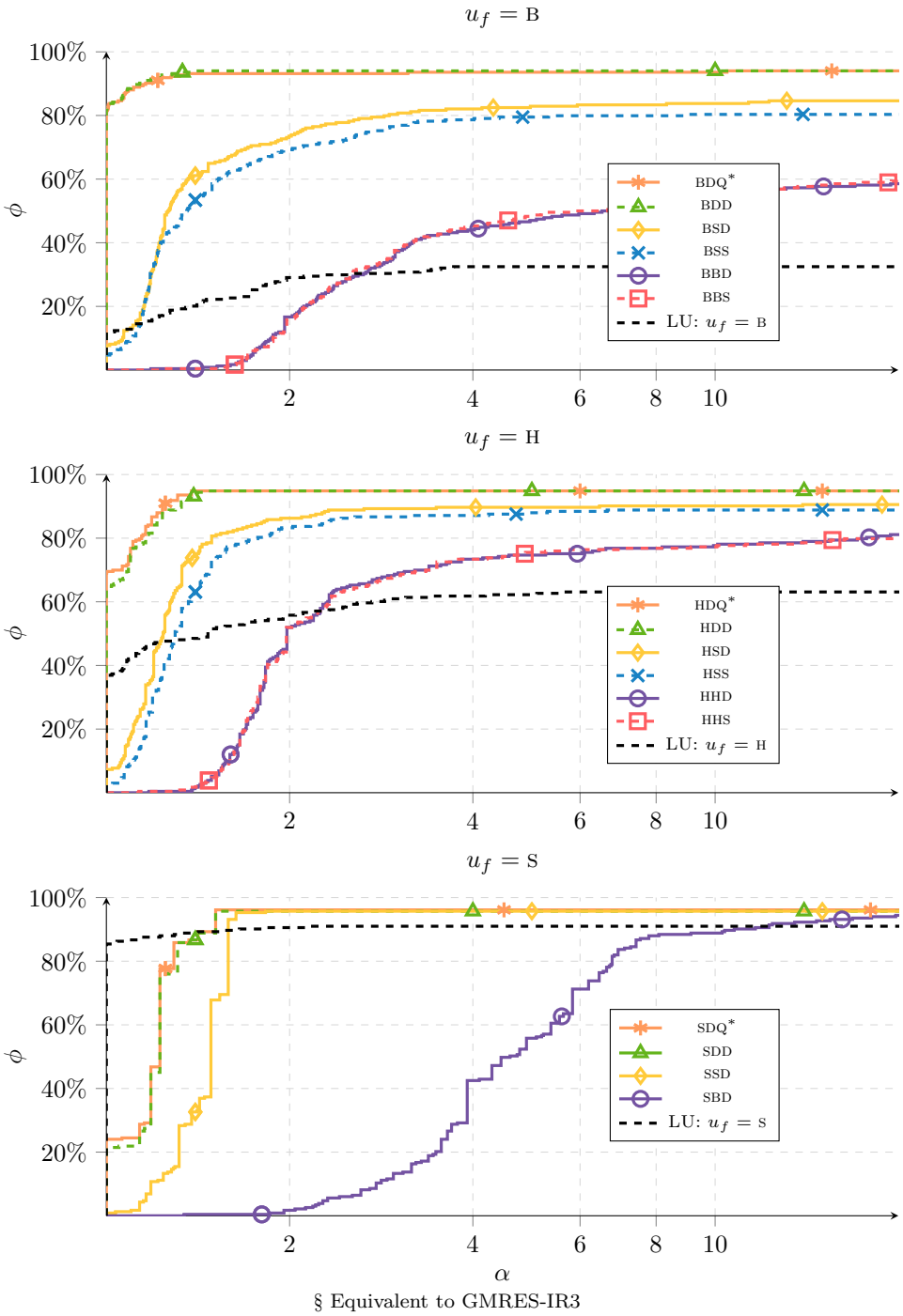
use the diagonal scaling method of [21], which first normalizes every row and column by its maximum value (preventing overflow), and then scales the matrix by a quantity λ close to the maximum representable value (to minimize underflow).

One difficulty is that the number of iterations is very sensitive to the choice of the GMRES stopping criterion τ and the scaling factor λ, and that the optimal choice of τ and λ is different for each variant and matrix. For the comparison to be as fair as possible, for each variant and each matrix, we have tested eight different values of τ (10^{-10}, 10^{-8}, 10^{-6}, 10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}, and 5×10^{-1}) and five different values of λ (10^4, 10^3, 10^2, 10^1, and 10^0), and taken the values that leads to the lowest number of iterations. We note in passing that the parameter τ could be taken into account in the error analysis. Indeed, τ provides a bound on the backward error (3.15) of the solution of the system $\widetilde{A}d_i = s_i$ by GMRES, which readily affects the convergence conditions.

We present in Figure 4.3 performance profiles for u_f = B, H, S, which plot the percentage ϕ of matrices for which a given variant converges in less than α times the number of LU solves required by the best variant. We also provide detailed results on a representative sample of these matrices for the case u_f = B in Table 4.1.

These results are in agreement with our theoretical study and with the previous experiments of section 4.1, and provide a confirmation of the relative robustness of each variant on a dataset of real-life applications matrices. For example, with u_f = B, LU-IR3 only converges for about 35% of the matrices, whereas with GMRES-IR3 (BDQ variant), we are able to process about 95% of the matrices (the 5% remaining being highly ill-conditioned matrices). Crucially, the new GMRES-IR5 variants (with relaxed u_p and/or u_g) are all more robust than LU-IR3, converging for a much larger percentage of matrices. In particular, on this set of matrices, the BDD variant is as robust as BDQ and requires the same number of LU solves. Therefore, these experiments show that, in practice, we can switch from u_p = Q to u_p = D with no impact on the convergence of GMRES-IR.

The other two performance profiles with u_f = H and S show similar trends. Unsurprisingly, as we increase the precision of the factorization, LU-IR3 is able to converge on a wider range of matrices and so the range of matrices where GMRES-IR is relevant is narrower. This is especially the case for an fp32 factorization, where LU-IR3 is able to converge on almost 90% of the matrices. Note that this observation heavily depends on the dataset: for these matrices, the distribution of the $\kappa(A)$ is not



§ Equivalent to GMRES-IR3

FIG. 4.3. *Performance profile of LU-IR3 and GMRES-IR5 variants (with u_f, u_g, u_p as specified in the legend) on* 230 *SuiteSparse matrices. ϕ (y-axis) indicates the percentage of matrices for which a given variant requires less than α (x-axis) times the number of LU solves required by the best variant. GMRES-IR5 variants with an asterisk "*" are equivalent to GMRES-IR3 variants.*

TABLE 4.1

Number of LU solves on a sample of SuiteSparse matrices for LU-IR3 and GMRES-IR5 variants with u_f = B. $\kappa(A)$ is the original condition number, $\kappa(A_s)$ is the condition number after scaling. The matrices are sorted by increasing $\kappa(A_s)$. A "—" indicates failure to converge. The GMRES-IR5 variant with an asterisk "" is equivalent to GMRES-IR3.*

Name	n	$\kappa(A)$	$\kappa(A_s)$	LU-IR3	BBS	BBD	BSS	BSD	BDD	BDQ*
dw256B	512	3.7E+00	1.8E+00	8	24	24	10	10	10	10
gre_512	512	1.6E+02	1.7E+02	16	28	28	17	17	14	14
mahindas	1258	2.1E+13	9.8E+02	12	34	36	14	12	12	12
dw1024	2048	2.1E+03	1.9E+03	18	36	36	20	20	19	19
fs_541_4	541	1.2E+10	6.9E+03	—	48	48	33	35	28	28
rajat12	1879	6.9E+05	8.1E+03	—	94	69	29	29	24	24
sherman1	1000	1.6E+04	9.3E+03	—	78	78	30	26	26	26
watt_2	1856	1.4E+11	3.0E+04	—	522	497	32	32	26	26
bp_600	822	1.5E+06	6.6E+04	—	56	54	23	25	19	19
bwm2000	2000	2.4E+05	2.0E+05	—	—	—	237	197	148	148
meg1	2904	1.4E+12	3.4E+05	13	28	26	17	17	14	14
lnsp_511	511	3.3E+15	5.8E+05	—	302	268	39	30	23	23
hangGlider_2	1647	1.4E+09	1.2E+06	—	52	56	32	33	31	31
tub1000	1000	1.3E+06	1.3E+06	—	—	—	354	254	114	114
1138_bus	1138	8.6E+06	2.4E+06	—	—	—	238	163	83	83
gre_1107	1107	3.2E+07	2.3E+07	—	—	—	244	185	75	75
rajat19	1157	1.1E+10	1.4E+08	—	—	—	—	—	70	69
spaceStation_7	1134	3.9E+11	1.7E+08	—	—	—	—	220	127	127
bcsstk19	817	1.3E+11	1.2E+09	—	—	—	—	—	307	209
reorientation_3	2513	1.5E+21	8.9E+11	—	—	—	—	—	144	188
fs_760_3	760	9.8E+19	2.6E+12	—	—	—	—	—	—	—
nnc1374	1374	3.7E+14	5.2E+12	—	—	—	—	—	85	92
lung1	1650	5.1E+19	1.1E+13	—	—	—	—	—	27	27

uniform and is mainly concentrated between 10^3 and 10^6, which explains why LU-IR3 performs well with an fp32 factorization.

A performance comparison of the actual runtime of the variants is outside our scope, but we can nevertheless extrapolate some performance trends based on Figure 4.3 and on the assumptions that (1) the LU solves dominate the overall runtime of the iterative phase of the solver; and (2) a bfloat16 LU solve is twice faster than an fp32 one, which is itself twice faster than an fp64 one. With this performance model and considering for example u_f = B, we can expect LU-IR3 (which uses bfloat16 LU solves) to be the fastest method as long as it does not require more than twice the number of LU solves of a variant with u_p = S, that is, for about 30% of the matrices. Similarly, the BSS variant should outperform BDD for over 70% of the matrices, making it the best variant for about $70 - 30 = 40\%$ of the matrices. Finally, on our set of matrices, the BDD variant never requires more LU solves than the BDQ one and therefore it should be the best variant for the remaining 30% of the matrices. In the next section we give a more precise model of the cost of GMRES-IR5.

5. Discussion on the complexity of GMRES-IR5. LU-IR3 and GMRES-IR5 improve the execution time over a direct solver by substantially reducing the number of high precision flops to obtain a solution of comparable accuracy. The execution time of these algorithms can be modeled using the following expressions:

$$
\begin{aligned}
\text{cost(LU-IR3)} &= \omega_f c_{\text{F}} + \#it_{\text{outer}} \times \Big(\omega_r(c_{\text{A}} + c_{\text{MV}}) + \omega_f c_{\text{LU}} + \omega_u c_{\text{A}}\Big), \\
\text{cost(GMRES-IR5)} &= \omega_f c_{\text{F}} \\
&\quad + \#it_{\text{outer}} \times \Big(\omega_r(c_{\text{A}} + c_{\text{MV}}) + \omega_p(c_{\text{A}} + c_{\text{MV}} + c_{\text{LU}}) + \omega_u c_{\text{A}}\Big) \\
&\quad + \#it_{\text{inner}} \times \Big(\omega_p(c_{\text{MV}} + c_{\text{LU}}) + \omega_g c_{\text{GMRES}}\Big),
\end{aligned}
$$

where the $\omega_{f,p,r,g,u}$ coefficients quantify the time for one arithmetic operation corresponding to the precision parameters u_f, u_p, u_r, u_g, u, $\#it_{\text{outer}}$ and $\#it_{\text{inner}}$ correspond to the number of outer (IR steps) and cumulated inner (GMRES) iterations, and c_{A}, c_{F}, c_{LU}, c_{MV}, and c_{GMRES} measure the flops cost of the vector–vector addition/subtraction, the LU factorization, the LU solve, the matrix–vector product with A, and all the operations in one inner GMRES iteration except the preconditioned matrix–vector product, respectively. In cost(GMRES-IR5), the term $\#it_{\text{outer}} \times (\omega_r(c_{\text{A}} + c_{\text{MV}}) + \omega_u c_{\text{A}})$ captures the cost of steps 4 and 6 in Algorithm 3.1. The term $\#it_{\text{outer}} \times \omega_p(c_{\text{A}} + c_{\text{MV}} + c_{\text{LU}})$ corresponds to the GMRES initialization overhead which computes the initial residual $\widehat{U}^{-1}\widehat{L}^{-1}Ad_{i,0} - \widehat{U}^{-1}\widehat{L}^{-1}r_i$ at each call of GMRES (step 5 of Algorithm 3.1), where $d_{i,0}$ is the initial guess or starting vector. The term $\#it_{\text{inner}} \times (\omega_p(c_{\text{MV}} + c_{\text{LU}}) + \omega_g c_{\text{GMRES}})$ covers the cost of the inner iterations of GMRES. Finally, the term $\omega_f c_{\text{F}}$ is the cost of the factorization which is computed once.

The actual value of these parameters, in practice, depends on numerous factors which are in general hard to predict. The c coefficients, in particular, will vary depending on the matrix being either dense or sparse and, in the latter case, on its structure. The ω coefficients, instead, depend on the properties of the software implementation and the efficiency of third party libraries (especially in a parallel setting), on the fact that some operations may include explicit or implicit data conversions, and on the features of the underlying hardware. Finally the number of outer and inner iterations depend on the numerical properties of the problem and are, in general, difficult to predict. It must also be noted that, in order to fully assess the performance of the proposed methods, a similar model must be defined also for the memory consumption. For all these reason, we will not use this model to draw conclusions on the behavior of the proposed methods; instead, we refer the reader to [10], [11], [12] for the dense case and [1] for the sparse case, where actual parallel implementations of these methods are presented and their performance evaluated on real-life problems.

6. Conclusions. We have addressed the solution of linear systems of equations by means of iterative refinement in mixed precision. The renewed interest in this method stems from the trend for computer hardware to support fast low precision floating-point arithmetic. Our baseline is the work by Carson and Higham [5], who propose a method, GMRES-IR3, that employs up to three precisions and in which a preconditioned mixed precision GMRES method is used for solving the correction equation in order to converge on badly conditioned problems. In its original form, this method requires twice the working precision to be used when applying the preconditioned matrix to a vector, which can be expensive, especially when the working precision is double precision. By relaxing the assumptions on the precision of the operations within the GMRES solver we have proposed a GMRES-based iterative refinement method with up to five different precisions in play, which we call GMRES-IR5.

We extended the rounding error analysis of Carson and Higham to cover GMRES-IR5. As a key component of this analysis, we extended the work of Paige et al. [27] to prove the backward stability of a mixed precision GMRES method. Based on this result we derived conditions on $\kappa(A)$ that guarantee the convergence of GMRES-IR5.

Our results show that GMRES-IR5 can accurately and reliably solve relatively badly conditioned problems in potentially lower time and memory than GMRES-IR3 thanks to the use of lower precision arithmetic in the GMRES iterations. Although the combined use of multiple precisions results in hundreds or thousands of different variants (depending on how many precisions are available on the system), we provided a list of rules that we used to identify a much smaller set of practical interest; these include variants for which only an experimental evaluation was available prior to this work. Finally, we presented an experimental analysis of the GMRES-IR5 method on randomly generated and SuiteSparse matrices. The experimental results are in good agreement with our theoretical analysis and show that GMRES-IR5 is a robust and versatile method for solving linear systems of equations.

REFERENCES

[1] 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. Softw., 49 (2023), https://doi.org/10.1145/3582493, https://doi.org/10.1145/3582493.

[2] C. Boutsikas, P. Drineas, and I. C. F. Ipsen, *Small singular values can increase in lower precision*, 2023, https://arxiv.org/abs/2303.03547.

[3] A. Buttari, J. Dongarra, J. Langou, J. Langou, P. Luszczek, and J. Kurzak, *Mixed precision iterative refinement techniques for the solution of dense linear systems*, International Journal of High Performance Computing Applications, 21 (2007), https://doi.org/10.1177/1094342007084026.

[4] 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.

[5] 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.

[6] 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.

[7] M. P. Connolly, N. J. Higham, and T. Mary, *Stochastic rounding and its probabilistic backward error analysis*, SIAM J. Sci. Comput., 43 (2021), pp. A566–A585, https://doi.org/10.1137/20m1334796.

[8] *cuSOLVER: Direct Linear Solvers on NVIDIA GPUs*. https://developer.nvidia.com/cusolver.

[9] J. W. Demmel, *The probability that a numerical analysis problem is difficult*, Mathematics of Computation, 50 (1988), pp. 449–480, http://www.jstor.org/stable/2008617.

[10] A. Haidar, A. Abdelfattah, M. Zounon, P. Wu, S. Pranesh, S. Tomov, and J. Dongarra, *The design of fast and energy-efficient linear solvers: On the potential of half-precision arithmetic and iterative refinement techniques*, in Computational Science—ICCS 2018, Y. Shi, H. Fu, Y. Tian, V. V. Krzhizhanovskaya, M. H. Lees, J. Dongarra, and P. M. A. Sloot, eds., Springer, Cham, Switzerland, 2018, pp. 586–600, https://doi.org/10.1007/978-3-319-93698-7_45.

[11] A. Haidar, H. Bayraktar, S. Tomov, J. Dongarra, and N. J. Higham, *Mixed-precision iterative refinement using tensor cores on GPUs to accelerate solution of linear systems*, Proc. Roy. Soc. London Ser. A, 476 (2020), p. 20200110, https://doi.org/10.1098/rspa.2020.0110.

[12] A. Haidar, S. Tomov, J. Dongarra, and N. J. Higham, *Harnessing GPU tensor cores for fast FP16 arithmetic to speed up mixed-precision iterative refinement solvers*, in Proceedings of the International Conference for High Performance Computing, Networking, Storage, and Analysis, SC18 (Dallas, TX), Piscataway, NJ, USA, 2018, pp. 47:1–47:11, https://doi.org/10.1109/SC.2018.00050.

[13] D. J. Higham, N. J. Higham, and S. Pranesh, *Random matrices generating large growth in LU factorization with pivoting*, SIAM J. Matrix Anal. Appl., 42 (2021), pp. 185–201, https://doi.org/10.1137/20M1338149.

[14] 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.

[15] N. J. Higham, *How fast is quadruple precision arithmetic?* https://nhigham.com/2017/08/31/how-fast-is-quadruple-precision-arithmetic/, June 2016.

[16] N. J. Higham, *Error analysis for standard and GMRES-based iterative refinement in two and three-precisions*, MIMS EPrint 2019.19, Manchester Institute for Mathematical Sciences, The University of Manchester, UK, Nov. 2019, http://eprints.maths.manchester.ac.uk/2735/.

[17] 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.

[18] N. J. Higham and T. Mary, *Sharper probabilistic backward error analysis for basic linear algebra kernels with random data*, SIAM J. Sci. Comput., 42 (2020), pp. A3427–A3446, https://doi.org/10.1137/20M1314355.

[19] N. J. Higham and T. Mary, *Mixed precision algorithms in numerical linear algebra*, MIMS EPrint 2021.20, Manchester Institute for Mathematical Sciences, The University of Manchester, UK, Dec. 2021, http://eprints.maths.manchester.ac.uk/2849/. Revised February 2022. To appear in Acta Numerica.

[20] N. J. Higham and S. Pranesh, *Exploiting lower precision arithmetic in solving symmetric positive definite linear systems and least squares problems*, SIAM J. Sci. Comput., 43 (2021), pp. A258–A277, https://doi.org/10.1137/19M1298263.

[21] N. J. Higham, S. Pranesh, and M. Zounon, *Squeezing a matrix into half precision, with an application to solving linear systems*, SIAM J. Sci. Comput., 41 (2019), pp. A2536–A2551, https://doi.org/10.1137/18M1229511.

[22] A. Kiełbasiński, *Iterative refinement for linear systems in variable-precision arithmetic*, BIT Numerical Mathematics, 21 (1981), pp. 97–103.

[23] J. Lee, G. D. Peterson, D. S. Nikolopoulos, and H. Vandierendonck, *Air: Iterative refinement acceleration using arbitrary dynamic precision*, Parallel Computing, 97 (2020), p. 102663, https://doi.org/https://doi.org/10.1016/j.parco.2020.102663, https://www.sciencedirect.com/science/article/pii/S0167819120300569.

[24] *Matrix algebra on GPU and multicore architectures (MAGMA)*. http://icl.cs.utk.edu/magma/.

[25] C. B. Moler, *Iterative refinement in floating point*, J. ACM, 14 (1967), p. 316–321, https://doi.org/10.1145/321386.321394, https://doi.org/10.1145/321386.321394.

[26] T. Ogita, *Accurate matrix factorization: Inverse LU and inverse QR factorizations*, SIAM J. Matrix Anal. Appl., 31 (2010), pp. 2477–2497, https://doi.org/10.1137/090754376.

[27] 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.

[28] S. M. Rump, *Inversion of extremely ill-conditioned matrices in floating-point*, Japan Journal of Industrial and Applied Mathematics, 26 (2009), pp. 249–277.

[29] B. Vieublé, *Mixed precision iterative refinement for the solution of large sparse linear systems*, PhD thesis, https://hal.science/tel-03975935.

[30] J. H. Wilkinson, *Rounding Errors in Algebraic Processes*, Notes on Applied Science No. 32, Her Majesty's Stationery Office, London, 1963. Also published by Prentice-Hall, Englewood Cliffs, NJ, USA. Reprinted by Dover, New York, 1994.