MANCHESTER
1824

Accelerating the Solution of Linear Systems by Iterative Refinement in Three Precisions

Carson, Erin and Higham, Nicholas J.

2017

ACCELERATING THE SOLUTION OF LINEAR SYSTEMS BY ITERATIVE REFINEMENT IN THREE PRECISIONS[*]

ERIN CARSON[†] AND NICHOLAS J. HIGHAM[‡]

Abstract. We propose a general algorithm for solving a $n \times n$ nonsingular linear system $Ax = b$ based on iterative refinement with three precisions. The working precision is combined with possibly different precisions for solving for the correction term and for computing the residuals. Via rounding error analysis of the algorithm we derive sufficient conditions for convergence and bounds for the attainable normwise forward error and normwise and componentwise backward errors. Our results generalize and unify many existing rounding error analyses for iterative refinement. With single precision as the working precision, we show that by using LU factorization in IEEE half precision as the solver and calculating the residuals in double precision it is possible to solve $Ax = b$ to full single precision accuracy for condition numbers $\kappa_2(A) \leq 10^4$, with the $O(n^3)$ part of the computations carried out entirely in half precision. We show further that by solving the correction equations by GMRES preconditioned by the LU factors the restriction on the condition number can be weakened to $\kappa_2(A) \leq 10^8$, although in general there is no guarantee that GMRES will converge quickly. Taking for comparison a standard $Ax = b$ solver that uses LU factorization in single precision, these results suggest that on architectures for which half precision is efficiently implemented it will be possible to solve certain linear systems $Ax = b$ up to twice as fast *and* to greater accuracy. Analogous results are given with double precision as the working precision.

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

AMS subject classifications. 65G50, 65F10,

1. Introduction. Iterative refinement is a method for improving an approximate solution y to a linear system $Ax = b$ by computing the residual $r = b - Ay$, solving the correction equation $Ad = r$, forming the update $y \leftarrow y + d$, and repeating these steps as necessary. We consider a general iterative refinement algorithm that includes a variety of existing ones as special cases. The algorithm contains three precisions:

- u is the precision at which the data A, b and the solution x are stored (the working precision),
- u_f is the precision at which the factorization of A is computed,
- u_r is the precision at which residuals are computed.

The precisions are assumed to satisfy

$$(1.1) \qquad\qquad\qquad u_r \leq u \leq u_f.$$

The algorithm also contains a fourth precision:

- u_s is the precision at which the correction equation is (effectively) solved, with $u \leq u_s \leq u_f$.

Whereas u, u_f, and u_r are intended to be possibly different precisions supported by the computing environment (ideally the hardware), u_s is essentially a parameter that

describes how accurately the correction equation is solved; it will take the value u_f or u in the cases of interest. The precise meaning of u_s is explained below.

ALGORITHM 1.1. *Let the nonsingular matrix $A \in \mathbb{R}^{n \times n}$ and $b \in \mathbb{R}^n$ be given in precision u. This algorithm uses iterative refinement to generate a sequence of approximations x_i, all stored in precision u, to the solution of $Ax = b$.*

1 Solve $Ax_0 = b$ in precision u_f and store x_0 at precision u.
2 for $i = 0: \infty$
3   Compute $r_i = b - Ax_i$ at precision u_r and round r_i to precision u_s.
4   Solve $Ad_i = r_i$ at precision u_s and store d_i at precision u.
5        $x_{i+1} = x_i + d_i$ at precision u.
6 end

Note that a different solver can be used on step 1 than on step 4. In practice, these solvers will be related, so although the precision u_f does not appear in the loop, information computed in step 1 at precision u_f will be used in step 4.

Algorithm 1.1 includes as special cases both old and more recent forms of iterative refinement, as we now explain.

In traditional iterative refinement the solver is LU factorization and residuals are computed at twice the working precision, which corresponds to $u_f = u_s = u$ and $u_r = u^2$. This form of iterative refinement was programmed by Wilkinson in 1948 [35] and used at that time by Wilkinson and his colleagues on desk calculating machines and Hollerith punched card machines [13], [37]. It was in common use up until the 1970s, owing to the fact that inner products could be cheaply accumulated at twice the working precision on many computers of that era. Early error analyses are given by Wilkinson for fixed point arithmetic [36] and Moler [26] for floating point arithmetic.

In fixed precision refinement all computations are at the same precision: $u_f = u_s = u = u_r$. This form of refinement started to be considered in the 1970s and was analyzed by Jankowski and Woźniakowski [21] for a general solver and by Skeel [31] for LU factorization. Higham [16] extended Skeel's analysis to a general solver and in [17] gave a further extension to allow for residuals computed in extra precision. LAPACK [3] implements fixed precision iterative refinement.

In the 2000s iterative refinement attracted renewed interest because on modern computer architectures single precision arithmetic is usually at least twice as fast as double precision arithmetic. With the base precision u equal to double precision, u_f and u_s equal to single precision, $u_r = u$, and a solver based on LU factorization, the most expensive part of the computation is done entirely in single precision. This usage is proposed and analyzed by Langou et al. [23] and has been exploited extensively by Dongarra and his coauthors; see [1, sec. 9] for a recent overview and further references.

Error analyses of the methods above all require A to be safely bounded away from singularity relative to the working precision. Carson and Higham [9] give a new forward error analysis of iterative refinement that identifies a mechanism that allows accurate solutions to be obtained to systems for which A has condition number as large as the order of the reciprocal of the unit roundoff. Their analysis requires the update equation in line 4 of Algorithm 1.1 to be solved with relative error less than 1, and they achieve this by the use of GMRES preconditioned with the LU factors.

Table 1.1 summarizes the main existing floating point error analyses. In the table we categorize a forward error bound as componentwise if it employs the condition numbers $\text{cond}(A)$ or $\text{cond}(A, x)$ defined in (2.2) below and normwise if it only employs $\kappa(A)$ defined in (2.1). A backward error analysis is classed as componentwise or

TABLE 1.1

Summary of existing rounding error analyses for iterative refinement in floating point arithmetic indicating (a) whether the analyses apply to LU factorization only or to an arbitrary solver, (b) whether the backward or forward error analyses are componentwise ("comp") or normwise ("norm"), and (c) the assumptions on the precisions u_f, u_s, u, u_r in Algorithm 1.1 ($u_f = u$ and $u_s = u_f$ unless otherwise stated).

	Year	Solver	Forward error	Backward error	Precisions
Moler [26]	1967	LU	norm	–	$u \geq u_r$
Stewart [33]	1973	LU	norm	–	$u \geq u_r$
Jankowski et al. [21]	1977	arb.	norm	norm	$u = u_r$
Skeel [31]	1980	LU	comp	comp	$u \geq u_r$
Higham [16]	1991	arb.	comp	comp	$u = u_r$
Higham [17], [18]	1997	arb.	comp	comp	$u \geq u_r$
Tisseur [34]	2001	arb.	norm	norm	$u \geq u_r$
Langou et al. [23]	2006	LU	norm	norm	$u_f \geq u = u_r$
Carson and Higham [9]	2017	arb.	comp	–	$u \geq u_r$
This work	2017	arb.	comp	comp, norm	$u_f \geq u_s \geq u \geq u_r$

TABLE 1.2

Parameters for four IEEE arithmetic precisions.

Type	Size	Range	Unit roundoff u
half	16 bits	$10^{\pm 5}$	$2^{-11} \approx 4.9 \times 10^{-4}$
single	32 bits	$10^{\pm 38}$	$2^{-24} \approx 6.0 \times 10^{-8}$
double	64 bits	$10^{\pm 308}$	$2^{-53} \approx 1.1 \times 10^{-16}$
quadruple	128 bits	$10^{\pm 4932}$	$2^{-113} \approx 9.6 \times 10^{-35}$

normwise according as it bounds the absolute value or the norm of the residual vector (we do not regard a componentwise analysis as implying a normwise one, as simply taking norms in a componentwise bound does not necessarily yield the strongest normwise result).

Half precision (16 bit) floating point arithmetic, defined as a storage format in the 2008 revision of the IEEE standard [20], is now starting to become available in hardware, for example in the NVIDIA P100 and V100 GPUs and the AMD Radeon Instinct MI25 GPU, on which it runs twice as fast as single precision arithmetic with a proportional saving in energy consumption. It is therefore now of interest to consider iterative refinement with u_f corresponding to half precision. Table 1.2 summarizes key parameters for four IEEE arithmetic precisions. Table 1.3 presents another view of iterative refinement, from the point of view of different ways in which it can be implemented in hardware-based IEEE arithmetic. We note that another context where low precision arithmetic is of interest is the hybrid iterative refinement method of Douglas, Mandel, and Miranker [11], which solves the correction equation using low-precision analog circuitry and computes residuals using higher-precision digital circuitry and effectively has $u_r = u$ and $u_f > u$.

The goal of this work is to develop new iterative refinement algorithms based on three precisions. We show that by using LU factorization in IEEE half precision as the solver, single precision for the working precision, and double precision for the computation of the residuals, it is possible to solve $Ax = b$ to full single precision accuracy for condition numbers $\kappa_2(A) \leq 10^4$, with the $O(n^3)$ part of the computations carried out entirely in half precision. We show further that by using GMRES preconditioned by the LU factors as the solver on step 4, the restriction on the condition number can be weakened to $\kappa_2(A) \leq 10^8$. These results provide the ability to solve $Ax = b$ at up

TABLE 1.3

Different scenarios for iterative refinement in IEEE arithmetic. The columns represent different choices for u_f, u, and u_r, where in the notation of Algorithm 1.1 the data is stored at precision u, the solves in steps 1 and 4 are carried out in precision $u_f = u_s$, and residuals are computed at precision u_r. The last column indicates whether any existing backward or forward error analysis is applicable to this situation when LU factorization is used as the solver.

Usage	Precision			Existing analysis?
	Half	Single	Double	
Traditional		data, solve	residual	√
Traditional			data, solve, residual	√
2000s		solve	data, residual	√
New	solve	data, residual		√
New	solve	data	residual	×
New	solve		data, residual	√

to twice the speed and the same accuracy as by traditional iterative refinement with LU factorization in single precision and double precision residuals.

In order to understand the behavior of the new algorithms we provide a thorough rounding error analysis of Algorithm 1.1 in its full generality. In doing so, we

- provide rigorous componentwise forward error bounds and both componentwise and normwise backward error bounds,
- make minimal assumptions about the solvers used in steps 1 and 4, so that the analysis is applicable to all the situations mentioned above, as well as to others that can be envisaged,
- treat the precisions u_f, u_s, u, and u_r as independent parameters.

Our results generalize and unify most existing analyses, including the recent forward error analysis of [9]. We make one omission: we do not try to prove a "one step of iterative refinement in fixed precision implies componentwise backward stability" result [16], [31], which is of lesser practical importance. However, such a result can be obtained by extending our analysis, under further assumptions on the solver.

Iterative refinement is often used as a way to restore stability when a factorization has been computed in a way that preserves structure and reduces cost at the expense of potential stability, as for example with sparse matrices [7], [38] or symmetric quasidefinite matrices [14]. Our analysis can be applied to these situations, since we make very general assumptions on the solver.

Our attention is focused exclusively on iterative refinement as described in Algorithm 1.1. We do not consider here recursive (also known as binary cascade, or k-fold) iterative refinement, in which each solve on step 4 of the algorithm is carried out by a recursive application of iterative refinement, possibly with increasing precision [22], [32]. We also do not consider hybrid schemes, such as that in [8], that combine iterative refinement with some other iterative method in such a way that the basic structure of the algorithm is changed.

2. Preliminaries. We now summarize our notation and our assumptions on the solver. We will use the standard model of floating point arithmetic [18, sec. 2.2]. Given an integer k, we define

$$\gamma_k = ku/(1 - ku).$$

A superscript on γ will denote that u carries that superscript as a subscript; thus, for example, $\gamma_k^r = ku_r/(1 - ku_r)$.

For a nonsingular matrix A and a vector x, we need the normwise condition number

$$\kappa_\infty(A) = \|A^{-1}\|_\infty \|A\|_\infty \tag{2.1}$$

and the componentwise condition numbers

$$\operatorname{cond}(A) = \| \, |A^{-1}||A| \, \|_\infty, \quad \operatorname{cond}(A,x) = \frac{\| \, |A^{-1}||A||x| \, \|_\infty}{\|x\|_\infty}, \tag{2.2}$$

where $|A| = (|a_{ij}|)$. These condition numbers measure the sensitivity of the solution of $Ax = b$ to small normwise and componentwise perturbations, respectively [18, chap. 7].

Inequalities between vectors or matrices are interpreted componentwise. We denote by $fl_r(\cdot)$ the evaluation of the argument of fl_r in precision u_r.

We assume that the solver used in step 4 of Algorithm 1.1 produces a computed solution $\widehat{d_i}$ to $Ad_i = \widehat{r_i}$ satisfying three conditions:

$$\widehat{d_i} = (I + u_s E_i)d_i, \quad u_s\|E_i\|_\infty < 1, \tag{2.3}$$

$$\|\widehat{r_i} - A\widehat{d_i}\|_\infty \le u_s(c_1\|A\|_\infty\|\widehat{d_i}\|_\infty + c_2\|\widehat{r_i}\|_\infty), \tag{2.4}$$

$$|\widehat{r_i} - A\widehat{d_i}| \le u_s G_i|\widehat{d_i}|, \tag{2.5}$$

where E_i, c_1, c_2, and G_i are functions of n, A, $\widehat{r_i}$, and u_s. The first assumption simply says that the normwise relative error $\|\widehat{d_i} - d_i\|_\infty/\|d_i\|_\infty$ is bounded by a multiple of u_s and is less than 1. The second assumption says that the normwise relative backward error $\eta(\widehat{d_i})$ is of order at most $\max(c_1, c_2)u_s$, where for an approximate solution y to $Ax = b$, $\eta(y)$ is given by [18, Thm. 7.1], [30]

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

The third condition will be needed in order to analyze the componentwise relative backward error, which for an approximate solution y to $Ax = b$ is given by [18, Thm. 7.3], [29]

$$\omega(y) := \min\{\epsilon : (A + \Delta A)y = b + \Delta b, \ |\Delta A| \le \epsilon|A|, \ |\Delta b| \le \epsilon|b|\} \tag{2.6}$$

$$= \max_i \frac{|b - Ay|_i}{(|A||y| + |b|)_i}, \tag{2.7}$$

where $\xi/0$ is interpreted as zero if $\xi = 0$ and infinity otherwise. Table 2.1 summarizes the sizes of c_1, c_2, E_i, and G_i in (2.3)–(2.5) for the solvers that will be considered in sections 7 and 8.

We present the rounding error analysis of Algorithm 1.1 in the next three sections, which provide forward error bounds, normwise backward error bounds, and componentwise bounds, respectively. The importance of scaling to avoid underflow and overflow when half precision is used is explained in section 6. In section 7 we specialize the results to the case where the solver is LU factorization and explain the numerical properties of iterative refinement in three precisions: half, single, and double. In section 8 we use GMRES preconditioned by the LU factors as the solver

	$u_s\|E_i\|_\infty$	$u_s \max(c_1, c_2)$	$u_s\|G_i\|_\infty$
IR w/LU fact.	$3nu_f\||A^{-1}||\widehat{L}||\widehat{U}|\|_\infty$	$3nu_f\dfrac{\||\widehat{L}||\widehat{U}|\|_\infty}{\|A\|_\infty}$	$3nu_f\||\widehat{L}||\widehat{U}|\|_\infty$
GMRES-IR	$uf(n)(1+\gamma_n^f\kappa_\infty(A))^2$	$O(u)$	$O(u\|A\|_\infty)$

and show that the resulting algorithm is able to solve accurately a wider range of problems than algorithms whose solver is based on LU factorization. In section 9 we compare some of our new forms of iterative refinement with the single precision–double precision form proposed by by Langou et al. [23]. Numerical experiments are given in section 10 that test the predictions of the analysis. Conclusions are given in section 11.

3. Forward error analysis. For our forward error analysis of Algorithm 1.1 we will need the following lemma, which we state for a general p-norm ($1 \le p \le \infty$).

LEMMA 3.1. *Let* $w, z \in \mathbb{R}^n$. *Then* $w = Cz$ *for a matrix* $C \in \mathbb{R}^{n\times n}$ *with* $\|C\|_p = \|w\|_p/\|z\|_p$.

Proof. Let g be the vector dual to z with respect to the p-norm, so that $g^T z = 1$ and $\|g\|_q\|z\|_p = 1$, where $1/p + 1/q = 1$ [18, sec. 6.1]. Setting $C = wg^T$, we have $Cz = w$ and $\|C\|_p = \|w\|_p\|g\|_q = \|w\|_p/\|z\|_p$. □

In the analysis we will need to bound the vector $b - A\widehat{x}_i = A(x - \widehat{x}_i)$ in terms of $x - \widehat{x}_i$. We can write

$$(3.1) \qquad \|A(x - \widehat{x}_i)\|_\infty = \mu_i\|A\|_\infty\|x - \widehat{x}_i\|_\infty,$$

where μ_i satisfies

$$\kappa_\infty(A)^{-1} \le \mu_i \le 1.$$

We also have the componentwise bound

$$|A(x - \widehat{x}_i)| \le |A||x - \widehat{x}_i|.$$

We can capture both inequalities by writing

$$b - A\widehat{x}_i = A(x - \widehat{x}_i) = C_i(x - \widehat{x}_i),$$

where

$$(3.2a) \qquad |C_i| \le |A|,$$

$$(3.2b) \qquad \|C_i\|_\infty = \frac{\|A(x - \widehat{x}_i)\|_\infty}{\|x - \widehat{x}_i\|_\infty} = \mu_i\|A\|_\infty,$$

the equalities following from Lemma 3.1 and (3.1). By using the matrices C_i we can derive a general result that can be expressed in terms of either componentwise or normwise condition numbers.

Consider first the computation of r_i. There are two stages. First, $\widehat{s}_i = fl_r(b - A\widehat{x}_i) = b - A\widehat{x}_i + \Delta s_i$ is formed in precision u_r, so that

$$|\Delta s_i| \le \gamma_p^r(|b| + |A||\widehat{x}_i|),$$

where p is the maximum number of nonzeros in any row of $[A\ b]$ [18, sec. 3.5]; thus $p = n + 1$ for a dense matrix A and vector b. Second, the residual is rounded to precision u_s, so $\widehat{r}_i = fl_s(\widehat{s}_i) = \widehat{s}_i + f_i$, where $|f_i| \le u_s|\widehat{s}_i|$. Hence

$$(3.3) \qquad \widehat{r}_i = b - A\widehat{x}_i + \Delta r_i, \quad |\Delta r_i| \le u_s|b - A\widehat{x}_i| + (1 + u_s)\gamma_p^r(|b| + |A||\widehat{x}_i|).$$

We rewrite the bound for Δr_i as

$$(3.4) \qquad\qquad |\Delta r_i| \le u_s|C_i||x - \widehat{x}_i| + (1 + u_s)\gamma_p^r(|b| + |A||\widehat{x}_i|),$$

where C_i satisfies (3.2).

For step 4 of Algorithm 1.1 we have, by (2.3),

$$\begin{aligned}
\widehat{d}_i - A^{-1}\widehat{r}_i &= u_s E_i A^{-1}\widehat{r}_i \qquad (u_s\|E_i\|_\infty \le 1)\\
&= u_s E_i A^{-1}(b - A\widehat{x}_i + \Delta r_i)\\
&= u_s E_i(x - \widehat{x}_i + A^{-1}\Delta r_i).
\end{aligned}$$

Hence, using (3.4),

$$|\widehat{d}_i - A^{-1}\widehat{r}_i| \le u_s|E_i|\big(|x - \widehat{x}_i| + |A^{-1}|\big[u_s|C_i||x - \widehat{x}_i| + (1 + u_s)\gamma_p^r(|b| + |A||\widehat{x}_i|)\big]\big)$$
$$(3.5) \qquad \le u_s|E_i|(I + u_s|A^{-1}||C_i|)|x - \widehat{x}_i| + u_s(1 + u_s)\gamma_p^r|E_i||A^{-1}|(|b| + |A||\widehat{x}_i|).$$

For step 5, using the variant [18, eq. (2.5)] of the rounding error model we have

$$(3.6) \qquad\qquad \widehat{x}_{i+1} = \widehat{x}_i + \widehat{d}_i + \Delta x_i, \qquad |\Delta x_i| \le u|\widehat{x}_{i+1}|.$$

Hence, using (3.3),

$$\begin{aligned}
\widehat{x}_{i+1} &= \widehat{x}_i + A^{-1}\widehat{r}_i + \widehat{d}_i - A^{-1}\widehat{r}_i + \Delta x_i\\
(3.7) \qquad &= x + A^{-1}\Delta r_i + (\widehat{d}_i - A^{-1}\widehat{r}_i) + \Delta x_i.
\end{aligned}$$

Therefore, by (3.4) and (3.5),

$$\begin{aligned}
|\widehat{x}_{i+1} - x| &\le u_s|A^{-1}||C_i||x - \widehat{x}_i| + (1 + u_s)\gamma_p^r|A^{-1}|(|b| + |A||\widehat{x}_i|)\\
&\quad + u_s|E_i|\big(I + u_s|A^{-1}||C_i|\big)|x - \widehat{x}_i|\\
&\quad + u_s(1 + u_s)\gamma_p^r|E_i||A^{-1}|(|b| + |A||\widehat{x}_i|) + u|\widehat{x}_{i+1}|\\
&= u_s\big((I + u_s|E_i|)|A^{-1}||C_i| + |E_i|\big)|x - \widehat{x}_i|\\
&\quad + (1 + u_s)\gamma_p^r(I + u_s|E_i|)|A^{-1}|(|b| + |A||\widehat{x}_i|) + u|\widehat{x}_{i+1}|.
\end{aligned}$$

We summarize the analysis in a theorem.

THEOREM 3.2. *Let Algorithm 1.1 be applied to a linear system $Ax = b$, where $A \in \mathbb{R}^{n\times n}$ is nonsingular, and assume the solver used on step 4 satisfies (2.3). For $i \ge 0$ the computed iterate $\widehat{x}_{i+1}$ satisfies*

$$|x - \widehat{x}_{i+1}| \le F_i|x - \widehat{x}_i| + f_i,$$

where

$$\begin{aligned}
F_i &= u_s(I + u_s|E_i|)|A^{-1}||C_i| + u_s|E_i|,\\
f_i &= (1 + u_s)\gamma_p^r(I + u_s|E_i|)|A^{-1}|(|b| + |A||\widehat{x}_i|) + u|\widehat{x}_{i+1}|
\end{aligned}$$

and

$$\text{(3.8)} \qquad \|F_i\|_\infty \le 2u_s \min\big(\text{cond}(A), \kappa_\infty(A)\mu_i\big) + u_s\|E_i\|_\infty,$$
$$\|f_i\|_\infty \le 2(1 + u_s)\gamma_p^r \||A^{-1}||A|(|x| + |\widehat{x}_i|)\|_\infty + u\|\widehat{x}_{i+1}\|_\infty.$$

We note that both terms are needed in the expression $\min(\text{cond}(A), \kappa_\infty(A)\mu_i)$ in (3.8). When A is diagonal, for example, the first term in the min is the smaller, since $\text{cond}(A) = 1$, whereas the second term is the smaller when $\text{cond}(A) \approx \kappa_\infty(A) \gg 1$ and $\mu_i \ll 1$.

We can now state a result about the convergence and attainable accuracy of Algorithm 1.1.

COROLLARY 3.3. *Under the conditions of Theorem 3.2, as long as*

$$\text{(3.9)} \qquad \phi_i = 2u_s \min\big(\text{cond}(A), \kappa_\infty(A)\mu_i\big) + u_s\|E_i\|_\infty$$

is sufficiently less than 1*, the forward error is reduced on the ith iteration by a factor approximately* ϕ_i *until an iterate* $\widehat{x}$ *is produced for which*

$$\text{(3.10)} \qquad \frac{\|x - \widehat{x}\|_\infty}{\|x\|_\infty} \lesssim 4pu_r \,\text{cond}(A, x) + u.$$

When $u_s = u$ this result can be compared with two earlier results for general solvers. It is stronger than [17, Thms. 3.1, 3.2], [18, Thms. 12.1, 12.2] because of the presence of the term $\kappa_\infty(A)\mu_i$, whose significance we will explain in section 8. It is also stronger than [9, Thm. 2.1], which does not have the $\text{cond}(A)$ term in (3.9).

Note that ϕ_i in (3.9) depends only on u_s. This means that the rate of convergence of iterative refinement depends only on the effective precision of the solves, and is not affected by the precision at which the data is stored or the residual is computed. By contrast, the limiting precision (3.10) depends on u_r and u, but not on u_s.

Note also that the precision u_f used for the initial solve in step 1 does not appear in Theorem 3.2 or Corollary 3.3. However, u_f does affect the accuracy of $\widehat{x}_0$ and hence the number of iterations required.

4. Normwise backward error analysis. Now we turn our attention to the behavior of the normwise backward error.

Multiplying (3.7) by A gives

$$\text{(4.1)} \qquad A\widehat{x}_{i+1} - b = \Delta r_i + A\widehat{d}_i - \widehat{r}_i + A\Delta x_i.$$

Writing $h_i = \widehat{r}_i - A\widehat{d}_i$, by assumption (2.4) we have

$$\|h_i\|_\infty \le u_s(c_1\|A\|_\infty\|\widehat{d}_i\|_\infty + c_2\|\widehat{r}_i\|_\infty)$$
$$\le u_s(c_1\|A\|_\infty\|A^{-1}\|_\infty(\|\widehat{r}_i\|_\infty + \|h_i\|_\infty) + c_2\|\widehat{r}_i\|_\infty),$$

which, assuming $c_1\kappa_\infty(A)u_s < 1$, gives

$$\|h_i\|_\infty \le u_s\frac{c_1\kappa_\infty(A) + c_2}{1 - c_1\kappa_\infty(A)u_s}\|\widehat{r}_i\|_\infty.$$

Hence, from (4.1), using (3.6) and two invocations of (3.3),

$$\|b - A\widehat{x}_{i+1}\|_\infty \leq u_s\|b - A\widehat{x}_i\|_\infty + (1 + u_s)\gamma_p^r(\|b\|_\infty + \|A\|_\infty\|\widehat{x}_i\|_\infty)$$
$$+ u_s\frac{c_1\kappa_\infty(A) + c_2}{1 - c_1\kappa_\infty(A)u_s}\|\widehat{r}_i\| + u\|A\|_\infty\|\widehat{x}_{i+1}\|_\infty$$
$$\leq u_s\left(1 + (1 + u_s)\frac{c_1\kappa_\infty(A) + c_2}{1 - c_1\kappa_\infty(A)u_s}\right)\|b - A\widehat{x}_i\|_\infty$$
$$+ \left(1 + \frac{u_s(c_1\kappa_\infty(A) + c_2)}{1 - c_1\kappa_\infty(A)u_s}\right)(1 + u_s)\gamma_p^r(\|b\|_\infty + \|A\|_\infty\|\widehat{x}_i\|_\infty)$$
$$+ u\|A\|_\infty\|\widehat{x}_{i+1}\|_\infty.$$

We summarize our findings in the next two results.

THEOREM 4.1. *Let Algorithm* 1.1 *be applied to a linear system* $Ax = b$ *with a nonsingular matrix* $A \in \mathbb{R}^{n \times n}$ *satisfying* $c_1\kappa_\infty(A)u_s < 1$, *and assume the solver used on step* 4 *satisfies* (2.4). *Then for* $i \geq 0$ *the computed iterate* $\widehat{x}_{i+1}$ *satisfies*

$$\|b - A\widehat{x}_{i+1}\|_\infty \leq \alpha_i\|b - A\widehat{x}_i\|_\infty + \beta_i,$$

where

$$\alpha_i = u_s\left(1 + (1 + u_s)\frac{c_1\kappa_\infty(A) + c_2}{1 - c_1\kappa_\infty(A)u_s}\right),$$
$$\beta_i = \left(1 + \frac{u_s(c_1\kappa_\infty(A) + c_2)}{1 - c_1\kappa_\infty(A)u_s}\right)(1 + u_s)\gamma_p^r(\|b\|_\infty + \|A\|_\infty\|\widehat{x}_i\|_\infty) + u\|A\|_\infty\|\widehat{x}_{i+1}\|_\infty.$$

COROLLARY 4.2. *Under the conditions of Theorem* 4.1, *if* $\phi = (c_1\kappa_\infty(A) + c_2)u_s$ *is sufficiently less than* 1 *then the residual is reduced on each iteration by a factor approximately* ϕ *until*

$$\|b - A\widehat{x}_i\|_\infty \lesssim \gamma_p^r(\|b\|_\infty + \|A\|_\infty\|\widehat{x}_{i-1}\|_\infty) + u\|A\|_\infty\|\widehat{x}_i\|_\infty.$$

Under the conditions of the corollary, making the reasonable assumption that $\|\widehat{x}_{i-1}\|_\infty \approx \|\widehat{x}_i\|_\infty$ and using $u_r \leq u$, we have, ultimately,

$$(4.2) \qquad \|b - A\widehat{x}_i\|_\infty \lesssim pu(\|b\|_\infty + \|A\|_\infty\|\widehat{x}_i\|_\infty).$$

In other words, $\eta(\widehat{x}_i) \lesssim pu$, that is, $\widehat{x}_i$ is a backward stable solution to the working precision.

Early error analyses of iterative refinement did not consider the residual, because when the solver is LU factorization with partial pivoting the residual of the original computed solution is already small, assuming there is no large element growth in the factorization. Starting with the work of Jankowski and Woźniakowski [21], it was appreciated that iterative refinement could cure instability in the solver, even in fixed precision. Our analysis shows this clearly: instability in the solver is captured by large values of c_1 and c_2 in (2.4), but as long as $(c_1\kappa_\infty(A) + c_2)u_s$ is sufficiently less than 1, Corollary 4.2 guarantees that iterative refinement will yield a small backward error.

Note that there is little or no benefit to the componentwise backward error of computing residuals at extra precision, since α_i in Theorem 4.1 is independent of u_r and the limiting residual is no smaller when $u_r < u$.

5. Componentwise backward error analysis. We now determine conditions under which Algorithm 1.1 achieves a small componentwise relative backward error.

From (3.6) we have

$$(5.1) \qquad |\widehat{x}_i| \le (1+u)|\widehat{x}_{i+1}| + |\widehat{d}_i|.$$

Using this inequality, along with assumption (2.5), (3.3), and (3.6) in (4.1), gives

$$|b - A\widehat{x}_{i+1}| \le u_s|b - A\widehat{x}_i| + (1+u_s)\gamma_p^r(1+u)(|b| + |A||\widehat{x}_{i+1}|)$$
$$(5.2) \qquad\qquad + Z_1|\widehat{d}_i| + u|A||\widehat{x}_{i+1}|,$$

where

$$(5.3) \qquad\qquad Z_1 = u_s G_i + (1+u_s)\gamma_p^r|A|.$$

From (2.5), using (3.3) and (5.1),

$$|\widehat{d}_i| \le |A^{-1}|(|\widehat{r}_i| + u_s G_i|\widehat{d}_i|)$$
$$\le |A^{-1}|\big[(1+u_s)|b - A\widehat{x}_i| + (1+u_s)\gamma_p^r(|b| + |A||\widehat{x}_i|) + u_s G_i|\widehat{d}_i|\big]$$
$$\le |A^{-1}|\big[(1+u_s)|b - A\widehat{x}_i| + (1+u_s)(1+u)\gamma_p^r(|b| + |A||\widehat{x}_{i+1}|)\big] + |A^{-1}|Z_1|\widehat{d}_i|.$$

Premultiplying by Z_1 gives

$$Z_1|\widehat{d}_i| \le Z_1|A^{-1}|\big[(1+u_s)|b - A\widehat{x}_i| + (1+u_s)(1+u)\gamma_p^r(|b| + |A||\widehat{x}_{i+1}|)\big] + Z_1|A^{-1}|Z_1|\widehat{d}_i|,$$

that is,

$$(I - Z_1|A^{-1}|)Z_1|\widehat{d}_i| \le Z_1|A^{-1}|\big[(1+u_s)|b - A\widehat{x}_i| + (1+u_s)(1+u)\gamma_p^r(|b| + |A||\widehat{x}_{i+1}|)\big].$$

Solving for $Z_1|\widehat{d}_i|$, we obtain

$$(5.4) \quad Z_1|\widehat{d}_i| \le M_1 Z_1|A^{-1}|\big[(1+u_s)|b - A\widehat{x}_i| + (1+u_s)(1+u)\gamma_p^r(|b| + |A||\widehat{x}_{i+1}|)\big],$$

where

$$(5.5) \qquad\qquad M_1 = (I - Z_1|A^{-1}|)^{-1},$$

and if we assume that $\|\,Z_1|A^{-1}|\,\|_\infty \le 1/2$ then M_1 exists, has nonnegative entries, and satisfies $\|M_1\|_\infty \le 2$. Substituting (5.4) in (5.2) gives

$$|b - A\widehat{x}_{i+1}| \le \big(u_s I + (1+u_s)M_1 Z_1|A^{-1}|\big)|b - A\widehat{x}_i|$$
$$+ (1+u_s)(1+u)\gamma_p^r(I + M_1 Z_1|A^{-1}|)(|b| + |A||\widehat{x}_{i+1}|) + u|A||\widehat{x}_{i+1}|.$$

We obtain the following result.

THEOREM 5.1. *Let Algorithm 1.1 be applied to a linear system $Ax = b$ with a nonsingular matrix $A \in \mathbb{R}^{n\times n}$ and assume the solver used on step 4 satisfies (2.5). Assume also that*

$$(5.6) \qquad u_s\|\,G_i|A^{-1}|\,\|_\infty + (1+u_s)\gamma_p^r\,\mathrm{cond}(A^{-1}) \le 1/2$$

for all i. Then for $i \ge 0$ the computed iterate $\widehat{x}_{i+1}$ satisfies

$$|b - A\widehat{x}_{i+1}| \le W_i|b - A\widehat{x}_i| + y_i,$$

where

$$W_i = u_s I + (1 + u_s) M_1 Z_1 |A^{-1}|,$$
$$y_i = (1 + u_s)(1 + u)\gamma_p^r (I + M_1 Z_1 |A^{-1}|)(|b| + |A||\widehat{x}_{i+1}|) + u|A||\widehat{x}_{i+1}|,$$

where Z_1 and M_1 are defined in (5.3) *and* (5.5), *respectively.*

To interpret the theorem, note first that $M_1 = I + O(u_s)$, so

$$W_i \lesssim u_s I + (u_s G_i + \gamma_p^r |A|)|A^{-1}|,$$

which implies

$$(5.7) \qquad \|W_i\|_\infty \lesssim u_s + u_s\| G_i |A^{-1}| \|_\infty + \gamma_p^r \operatorname{cond}(A^{-1}),$$

and so $\|W_i\|_\infty < 1$ in view of (5.6). Note that we can expect $G_i \geq |A|$ in practice, so the dominant term in this bound will be $u_s\| G_i |A^{-1}| \|_\infty$. We conclude that $\|W_i\|_\infty \ll 1$ if the solver is not too unstable and A is not too ill conditioned relative to precision u_s.

In the limit the y_i term dominates, but it is not a scalar multiple of $|A||\widehat{x}_{i+1}| + |b|$. This is not a problem if we wish to take norms and use this analysis to obtain a bound for the normwise backward error that exploits the more descriptive bound (2.5). In order to bound $\omega(\widehat{x}_{i+1})$ we need a simple lemma [17, Lem. 1.2].

LEMMA 5.2. *For $A \in \mathbb{R}^{n \times n}$ and $x \in \mathbb{R}^n$ we have*

$$|A||x| \leq \|A\|_\infty \xi(x)|x|,$$

where $\xi(x) = \max_j |x_j| / \min_j |x_j|$, with x_j denoting the jth component of x.

Using the lemma we have

$$|y_i| \lesssim \gamma_p^r \big(1 + \| (u_s G_i + \gamma_p^r |A|)|A^{-1}| \|_\infty\big)$$
$$\times \xi(|b| + |A||\widehat{x}_{i+1}|)(|b| + |A||\widehat{x}_{i+1}|) + u|A||\widehat{x}_{i+1}|.$$

Hence, ultimately, the componentwise relative backward error of x_{i+1} satisfies

$$(5.8) \qquad \omega(\widehat{x}_{i+1}) \lesssim \gamma_p^r \big(1 + u_s\| G_i |A^{-1}| \|_\infty + \gamma_p^r \operatorname{cond}(A^{-1})\big)\xi(|b| + |A||\widehat{x}_{i+1}|) + u.$$

This bound will be of order u as long as the solver is not too unstable, A is not too ill conditioned, and $\xi(|b| + |A||\widehat{x}_{i+1}|)$ is not too large. The latter condition essentially requires the vector $|b| + |A||x|$ to be not too badly scaled, which is a natural requirement, because when $|b| + |A||x|$ has a zero component the problem of computing a solution with a small componentwise relative backward error is ill posed; see [5], [18, p. 241].

Table 5.1 summarizes the sufficient conditions for convergence and the bounds on the limiting forward error and backward errors derived in this section and the previous two sections.

6. Scaling. Our analysis makes no assumptions on the floating point arithmetic other than that the three precisions obey the standard model and satisfy (1.1). As is usual in rounding error analyses we have ignored the possibility of underflow and overflow. In general, this is reasonable, but if we take u_f to be a low precision, specifically if u_f is IEEE half precision, then underflow or overflow is quite likely in step 4 of Algorithm 1.1, since the range of half precision numbers is only $10^{\pm 5}$.

Summary of the results of sections 3–5: conditions for convergence and the limiting size of the
forward error, normwise backward error, and componentwise backward error.

Error	Convergence condition	Bound on limiting value
Forward	$2u_s \min(\text{cond}(A), \kappa_\infty(A)\mu_i)$ $+ u_s\|E_i\|_\infty < 1$	$4pu_r \text{cond}(A,x) + u$
Normwise backward	$(c_1\kappa_\infty(A) + c_2)u_s < 1$	pu
Componentwise backward	$u_s\| \,|G_i|A^{-1}| \,\|_\infty + (1 + u_s)$ $\times \gamma_p^r \text{cond}(A^{-1}) \le 1/2$	$\gamma_p^r(1 + u_s\| \,|G_i|A^{-1}| \,\|_\infty + \gamma_p^r \text{cond}(A^{-1}))$ $\times \xi(|b| + |A||x|) + u$

In this case it is important to incorporate scaling in step 4. When the solver is LU factorization we can use any scheme for avoiding overflow in solving triangular systems. One such scheme is implemented in the LAPACK subroutine `xLATRS`, which performs scaling based on a coarse bound on the solution size to reliably avoid overflow [4], [10]. Here we propose a simple scaling that can be used with any solver, though it may not be optimal for any given solver. We replace steps 4 and 5 of Algorithm 1.1 by

4   $\theta = \|r_i\|_\infty$, $\widetilde{r}_i = r_i/\theta$. Solve $Ad_i = \widetilde{r}_i$ at precision u_s and store d_i at precision u.

5   $x_{i+1} = x_i + \theta d_i$ at precision u.

Since $\|\widetilde{r}_i\|_\infty = 1$, we have $1/\|A\|_\infty \le \|d_i\|_\infty \le \|A^{-1}\|_\infty$, so this scaling avoids the largest element of d_i overflowing or underflowing as long as $1/\|A\|_\infty$ does not underflow and $\|A^{-1}\|_\infty$ does not overflow.

7. Iterative refinement with LU factorization. We now explore the consequences of the results of our error analysis for the standard form of iterative refinement based on LU factorization. We recover known results and, in subsection 7.3, obtain new results for iterative refinement in three precisions.

Suppose that the solve on step 1 of Algorithm 1.1 is carried out by LU factorization with an appropriate form of pivoting and that the solves on step 4 are done with the LU factors. Throughout this section we take $u_s = u_f$. For notational simplicity we assume that any necessary row or column interchanges have been applied to A before the factorization starts, so that the factorization is $A = LU$.

Standard backward error analysis shows that the solution to $Ay = c$ computed at precision u_f satisfies

$$(7.1) \qquad (A + \Delta A)\widehat{y} = c, \quad |\Delta A| \le \gamma_{3n}^f |\widehat{L}||\widehat{U}|,$$

where $\widehat{L}$ and $\widehat{U}$ are the computed LU factors [18, Thm. 9.4]. Hence we can take $u_s|E_i| \approx 3nu_f|A^{-1}||\widehat{L}||\widehat{U}|$ in (2.3). Since, from [18, Thm. 9.3] it follows that $|A| \le (1 + \gamma_n^f)|\widehat{L}||\widehat{U}|$, the E_i term in (3.9) will dominate, and so in Corollary 3.3 we can take

$$(7.2) \qquad \phi \equiv \phi_i = 3nu_f\| \,|A^{-1}||\widehat{L}||\widehat{U}| \,\|_\infty \gtrsim 3nu_f \text{cond}(A).$$

Using [18, Lem. 9.6] it is possible to obtain a bound of the form $\phi \le f(n)\rho_n u_f \kappa_\infty(A)$, where f is a cubic polynomial and ρ_n is the growth factor for the LU factorization.

In order to be sure that iterative refinement converges, and does so at a reasonable rate, we need $\phi \ll 1$ and this is assured if A is not too ill conditioned and the factorization is not too unstable with respect to the precision u_f at which the factorization and substitutions are carried out.

We now consider three different scenarios for the three precisions u_f, u, and u_r.

7.1. Traditional refinement with residuals in extra precision. Consider the case where $u_f = u$ and $u_r = u^2$, which corresponds to traditional iterative refinement with residuals calculated at twice the working precision. Here, the limiting accuracy is, from (3.10),

$$(7.3) \qquad 4pu^2 \operatorname{cond}(A, x) + u \le 4pu \operatorname{cond}(A) \cdot u + u \le \left(\tfrac{4}{3}\phi + 1\right)u,$$

so as long as ϕ in (7.2) is sufficiently less than 1 we are assured of achieving a solution with normwise relative error of order u. We therefore recover a stronger version of the well known result first obtained by Moler [26], with $\operatorname{cond}(A)$ in place of $\kappa_\infty(A)$ in the convergence condition.

7.2. Fixed precision refinement. With $u_f = u$ and $u_r = u$ we have fixed precision iterative refinement, and ϕ is unchanged from the previous case. The difference is that the limiting accuracy is now

$$(7.4) \qquad 4pu \operatorname{cond}(A, x) + u \approx 4pu \operatorname{cond}(A, x).$$

Normwise and componentwise backward stability are shown by (4.2) and (5.8), under the conditions stated there. As originally shown by Skeel [31], the benefit of fixed precision iterative refinement for the forward error is that it gives a limiting accuracy of order $\operatorname{cond}(A, x)u$ instead of order $\kappa_\infty(A)u$ for the original computed solution, and this is irrespective of any instability in the factorization as long as $\phi \ll 1$ continues to hold.

7.3. Mixed precision refinement with lower precision solves. The third scenario of interest is where we compute the LU factorization and carry out the substitutions at less than the working precision. We consider four particular cases, all of which yield new results. We show the relevant choices of IEEE precisions, in the form "(u_f, u, u_r)."

Case 1: $u = u_r = u_f^2$ (half, single, single) or (single, double, double). This form of refinement has been analyzed and exploited by Langou et al. [23], and is also used by Arioli and Duff [6] and, for symmetric systems, by Hogg and Scott [19]. From (7.2), convergence is assured if

$$(7.5) \qquad \phi = 3n\| |A^{-1}||\widehat{L}||\widehat{U}| \|_\infty u^{1/2} < 1,$$

and assuming this condition holds the limiting accuracy is, from (3.10) and (7.5),

$$4pu \operatorname{cond}(A, x) + u \le u^{1/2} \cdot 4pu^{1/2} \operatorname{cond}(A) \lesssim \tfrac{4}{3}\phi u^{1/2} < \tfrac{4}{3}u^{1/2},$$

which is stronger than the limiting accuracy proportional to $\kappa_\infty(A)u$ obtained in [23]. Compared with fixed precision refinement we have a more stringent convergence requirement and the same limiting accuracy, but now the $O(n^3)$ flops part of the computation is done at precision $u^{1/2}$, which is a significant computational saving. The normwise and componentwise backward errors both reach order u under the assumptions that A is not too ill conditioned or the factorization too unstable with respect to precision $u^{1/2}$, and also that $|A||x| + |b|$ is not too badly scaled in the case of the componentwise backward error.

Case 2: $u_r = u^2$, $u = u_f^2$ (half, single, double) or (single, double, quad). Now we have *three* precisions in play—a case for which there is no existing analysis. Convergence is again assured if (7.5) holds, and if it does we now achieve a normwise relative error of order

$$4pu^2 \operatorname{cond}(A, x) + u \le u^{3/2} \cdot 4p \operatorname{cond}(A, x)u^{1/2} + u \lesssim \tfrac{4}{3}u^{3/2} + u \approx u.$$

TABLE 7.1
Different choices of IEEE standard precision for Algorithm 1.1 with LU factorization (assumed numerically stable) as the solver (and with $u_s = u_f$). The fourth column shows a bound on $\kappa_\infty(A)$ that must hold for the analysis to guarantee convergence (under suitable conditions described in the text) with limiting backward or forward errors of the orders shown in the final three columns.

				Backward error		
u_f	u	u_r	$\kappa_\infty(A)$	normwise	comp'wise	Forward error
half	single	single	10^4	single	single	$\text{cond}(A, x) \cdot 10^{-8}$
half	single	double	10^4	single	single	single
half	double	double	10^4	double	double	$\text{cond}(A, x) \cdot 10^{-16}$
half	double	quad	10^4	double	double	double
single	single	single	10^8	single	single	$\text{cond}(A, x) \cdot 10^{-8}$
single	single	double	10^8	single	single	single
single	double	double	10^8	double	double	$\text{cond}(A, x) \cdot 10^{-16}$
single	double	quad	10^8	double	double	double

Now we achieve full accuracy at precision u, albeit still only for problems with $\kappa_\infty(A)$ no larger than $u^{-1/2}$. Nevertheless, this is a significant gain over case 1 in return for a few residuals computed at precision u^2. The limiting backward errors are of order u, as in the previous case.

Case 3: $u = u_r = u_f^4$ (half, double, double). In this more extreme case the factorization is done at one quarter of the working precision. The convergence condition is, from (7.2), $\phi = 3n\| |A^{-1}||\widehat{L}||\widehat{U}| \|_\infty u^{1/4} < 1$, and the limiting accuracy is now, from (3.10),

$$4pu\,\text{cond}(A, x) + u \le u^{3/4} \cdot 4pu^{1/4}\,\text{cond}(A) + u \le \tfrac{4}{3}\phi u^{3/4} + u \lesssim u^{3/4}.$$

Again, the limiting backward errors are of order u.

Case 4: $u = u_f^4$, $u_r = u^2$ (half, double, quad). In this most extreme case the convergence condition is the same as in case 3, and the limiting accuracy is now $4pu^2\,\text{cond}(A, x) + u \lesssim u$. Again, the limiting backward errors are of order u.

Take u_f to be half precision. Case 2 shows that for a sufficiently well conditioned linear system $Ax = b$ with single precision data we can obtain the solution correct to single precision accuracy by doing only $O(n^2)$ operations in single or double precision with the dominant $O(n^3)$ part of the work at half precision. Case 3 shows that for double precision data the convergence condition is the same but the limiting accuracy is of order $\text{cond}(A, x)u$, and the computational saving over working entirely at precision u is even greater. In case 4 the limiting accuracy improves to u.

The statements about cost in this subsection assume that the number of required iterations is small and independent of n, which will be the case as long as ϕ in Corollary 3.3 is sufficiently less than 1.

We summarize the usages described in this section in Table 7.1.

8. Mixed precision refinement with preconditioned GMRES. Carson and Higham [9] introduce a new form of iterative refinement that corresponds to Algorithm 1.1 with $u_f = u$ and $u_r = u^2$ and a special way of solving the correction equation on line 4. The algorithm is intended to handle the situation where A is extremely ill conditioned, possibly even singular to working precision, so that $\kappa_\infty(A)$ could exceed u^{-1}. It computes an LU factorization $A \approx \widehat{L}\widehat{U}$ then solves the equation $Ad = \widehat{r}$ on step 4 by applying GMRES to the system

$$(8.1) \qquad \widetilde{A}d \equiv \widehat{U}^{-1}\widehat{L}^{-1}Ad = \widehat{U}^{-1}\widehat{L}^{-1}\widehat{r},$$

with all computations done at precision u except that the matrix–vector products with $\widetilde{A}$ needed by GMRES are evaluated at precision u^2. Carson and Higham give an error analysis similar to that in section 3 (but with $u_f = u_s = u$), making the key observation that μ_i in (3.1) is typically much less than 1 in the early stages of iterative refinement.

We now consider a more general GMRES-based algorithm involving three precisions rather than two. This is a special case of Algorithm 1.1 and we write it out in detail for clarity.

ALGORITHM 8.1 (GMRES-IR). *Let the nonsingular matrix $A \in \mathbb{R}^{n \times n}$ and $b \in \mathbb{R}^n$ be given in precision u. This algorithm uses GMRES-based iterative refinement using LU factors as preconditioners to generate a sequence of approximations x_i, all stored in precision u, to the solution of $Ax = b$.*

  1   Compute an LU factorization $A = LU$ in precision u_f.
  2   Solve $Ax_0 = b$ in precision u_f using the LU factors and store x_0 at precision u.
  3   for $i = 0 : \infty$
  4   Compute $r_i = b - Ax_i$ at precision u_r and round r_i to precision u.
  5   Solve $\widetilde{A}d_i \equiv \widehat{U}^{-1}\widehat{L}^{-1}Ad_i = \widehat{U}^{-1}\widehat{L}^{-1}r_i$ by GMRES at precision u, with
             matrix–vector products with $\widetilde{A}$ computed at precision u_r, and
             store d_i at precision u.
  6       $x_{i+1} = x_i + d_i$ at precision u.
  7   end

The analysis in [9, sec. 3] shows that if $u_r = u^2$ we can take in (2.3)

$$(8.2) \qquad u_s \|E\|_\infty \equiv uf(n)\kappa_\infty(\widetilde{A}),$$

where f is a quadratic polynomial and

$$(8.3) \qquad \kappa_\infty(\widetilde{A}) \leq \left(1 + \gamma_n^f \| \, |A^{-1}||\widehat{L}||\widehat{U}| \, \|_\infty\right)^2 \lesssim (1 + \gamma_n^f \kappa_\infty(A))^2,$$

these inequalities being pessimistic.

The reason for including the fourth precision u_s in Algorithm 1.1 is now clear: even though the LU factors in step 1 of Algorithm 8.1 are computed at precision u_f, the solve in step 5 that uses these factors achieves an error of order $u_s = u$. That the LU factors were computed at precision u_f is irrelevant to the preconditioned system, as long as the preconditioner $\widehat{L}\widehat{U}$ remains nonsingular. All that matters is that the factors yield an $\widetilde{A}$ with condition number much smaller than that of A.

The convergence condition $\phi_i \ll 1$ from the forward error analysis, where ϕ_i is defined in (3.9), therefore holds if

$$(8.4) \qquad 2u\kappa_\infty(A)\mu_i + f(n)u(1 + \gamma_n^f \kappa_\infty(A))^2 \ll 1.$$

As mentioned above, and explained in detail in [9], μ_i is much less than 1 in the early iterations, so this condition is in practice dominated by the second term, for which we need $f(n)u(\gamma_n^f)^2\kappa_\infty(A)^2 \ll 1$, and hence certainly $\kappa_\infty(A) < u^{-1/2}u_f^{-1}$; so (8.4) can hold for $\kappa_\infty(A)$ greater then u_f^{-1}. Then the limiting accuracy is, from (3.10), $4pu_r \operatorname{cond}(A, x) + u$. With $u_f = u$ and $u_r = u^2$ this reproduces the results of [9], giving a limiting accuracy of

$$(8.5) \qquad 4pu^2 \operatorname{cond}(A, x) + u \lesssim 4pu,$$

provided $\operatorname{cond}(A, x)u \leq 1$.

Different choices of IEEE precision for Algorithm 8.1 *The middle column shows a bound on* $\kappa_\infty(A)$ *that must hold for the analysis to guarantee convergence with the limiting backward or forward errors shown in the final three columns.*

u_f	u	u_r	$\kappa_\infty(A)$	Backward error normwise	comp'wise	Forward error
half	half	single	10^4	half	half	half
half	single	double	10^8	single	single	single
half	double	quad	10^{12} *	double	double	double
single	single	double	10^8	single	single	single
single	double	quad	10^{16}	double	double	double

———

* This bound is from the forward error analysis. The backward error analysis requires only $\kappa_\infty(A) \le 10^{16}$.

Now we set $u_f = u^{1/2}$ and $u_r = u^2$. Provided that

$$(8.6) \qquad 2u\kappa_\infty(A)\mu_i + f(n)u\big(1 + nu^{1/2}\kappa_\infty(A)\big)^2 \ll 1,$$

which, given the behavior of μ_i, essentially requires $\kappa_\infty(A) \ll u^{-1}$, Algorithm 8.1 will converge and achieve a limiting accuracy of (8.5). To be specific, this means that by taking u_f to be IEEE half precision, u single precision, and u_r double precision, we can potentially solve systems with $\kappa_\infty(A)$ possibly as large as u^{-1} to single precision accuracy while performing the LU factorization at half precision, so that only $O(n^2)$ of the flops are at single or double precision. We can go even further, by setting $u_f = u^{1/4}$ and $u_r = u^2$. Now (8.4) implies the condition $\kappa_\infty(A) \ll u^{-3/4}$ and gives a limiting accuracy of (8.5) again.

In order to achieve this potential we need the number of iterative refinement steps (outer iterations) and the number of iterations in the GMRES solves (inner iterations), each of which involves a matrix–vector product and two triangular solves in precision u_r, to be small. If GMRES takes $O(n)$ iterations to converge, each solve will require $O(n^3)$ operations in precision u_r, and so any potential savings from using a lower precision LU factorization will be lost. In the case of normal $\widetilde{A}$, the theoretical convergence rate of GMRES is completely determined by the spectrum of $\widetilde{A}$. While a small $\kappa_\infty(\widetilde{A})$ often corresponds to fast GMRES convergence, this is not always the case. For example, a cluster of eigenvalues close to the origin can cause stagnation of the GMRES residual until the nth iteration, regardless of the condition number of the matrix [24]. Since the GMRES convergence rate for normal A is well understood, this suggests potential strategies for improving the convergence rate in the event that a lower precision LU factorization causes slow GMRES convergence. We briefly discuss such strategies at the end of Section 10.

For nonnormal matrices, however, the convergence rate of GMRES is still not well understood, and the spectrum of $\widetilde{A}$ is irrelevant to the rate of GMRES convergence [15]. Nevertheless, our numerical experiments in Section 10.2.2 show that GMRES-IR in three precisions can be efficient even for ill-conditioned nonnormal matrices in some cases.

We also need to check the behavior of the residual for Algorithm 8.1. It is shown in [9] that the preconditioned system $\widetilde{A}d_i = \widehat{U}^{-1}\widehat{L}^{-1}r_i$, is solved with backward error of order u, and it is easy to show that this implies that the same is true of the original correction equation $Ad_i = r_i$, so that we can take c_1 and c_2 in (2.4) to be of order 1 and G_i to have norm of order $\|A\|_\infty$ in (2.5). It follows from Corollary 4.2 that

a normwise backward error of order u will be obtained if $\kappa_\infty(A)u$ is sufficiently less than 1. Similarly, the analysis of section 5 shows that a componentwise backward error of order u will be obtained if $\text{cond}(A)u$ is sufficiently less than 1, under the usual assumptions on the problem. In the case where $u_f = u$, both these conditions are much stricter than the condition (8.4) required for our forward error result— essentially because the backward error analysis is not able to exploit the behavior of the μ_i that is so favorable to the forward error analysis.

In Table 8.1 we summarize the practical usages of Algorithm 8.1 with IEEE arithmetic. The fourth line in the table corresponds to the algorithm proposed in [9]. The second line in the table summarizes our finding that the LU factorization can be computed in half precision instead of single precision and the algorithm will still obtain a result correct to single precision for $\kappa_\infty(A)$ up to 10^8—in other words, we can obtain as good a result at potentially half the cost.

Finally, we note that when u_f is half precision we could encounter overflow in computing x_0 on step 2, even if we scale as described in section 6. In this case we can simply set $x_0 = 0$. All our analysis is still valid, and when iterative refinement is rapidly converging this may have little effect on the number of iterations. As we mention in Section 10, we encounter this situation for one problem in our numerical experiments, when $(u_f, u, u_r) = (\text{half}, \text{single}, \text{quad})$ and $\kappa_\infty(A) > 10^{12}$.

9. Comparison with single–double implementations. Dongarra and his co-authors have made extensive use over the last decade of iterative refinement with double precision as the working precision and LU factorization computed in single precision as the solver, motivated by the fact that single precision arithmetic runs twice as fast as double precision arithmetic on modern architectures [1, sec. 9], [23]. Code implementing this form of refinement is available in the PLASMA library as routine `gesv` [2]. Table 9.1 shows how our new forms of iterative refinement compare with this approach. We see that:
- By changing to computing residuals in quadruple precision we can guarantee a forward error of order u (that is, the $\text{cond}(A, x)$ factor is removed).
- By reducing the precision of the LU factorization from single to half there is no loss in forward error or backward error, but the bound on $\kappa_\infty(A)$ for convergence to be guaranteed drops from 10^8 to 10^4.
- By switching to GMRES-IR and using quadruple precision residuals we can solve a larger class of problems ($\kappa_\infty(A)$ bounded by 10^{16}, or 10^{12} for the forward error bounds to hold when u_f is half precision) and are guaranteed a forward error of order u.

10. Numerical experiments. We have implemented Algorithms 1.1 and 8.1, with the scaling of section 6, in MATLAB version R2015b, using the built-in single and double precision arithmetics along with the `fp16` half-precision class written by Moler [25], [27].

In all the tests in this section we use dense matrices of order $n = 100$ generated by the built-in MATLAB function `gallery('randsvd',kappa,mode)` with specified 2-norm condition number `kappa`. Unless otherwise specified, we use the default mode 3, which generates a random matrix with geometrically distributed singular values. The right-hand sides b are generated as MATLAB `randn` vectors. For reproducibility, we issue the MATLAB function call `rng(1)` to set the random number generator seed before generating each problem A, b. We use the MATLAB `lu` function to compute the LU factorization with partial pivoting. For quadruple precision, we use the Advanpix

TABLE 9.1

Comparison of results for iterative refinement using the solvers given in the first column. The first row corresponds to the usage of Dongarra and his co-authors.

		u_f	u	u_r	$\kappa_\infty(A)$	Backward error normwise	comp'wise	Forward error
	LU	single	double	double	10^8	double	double	$\mathrm{cond}(A,x) \cdot 10^{-16}$
	LU	single	double	quad	10^8	double	double	double
	LU	half	double	double	10^4	double	double	$\mathrm{cond}(A,x) \cdot 10^{-16}$
	LU	half	double	quad	10^4	double	double	double
GMRES-IR	single	double	quad	10^{16}	double	double	double
GMRES-IR	half	double	quad	10^{12} *	double	double	double

* This bound is from the forward error analysis. The backward error analysis requires only $\kappa_\infty(A) \leq 10^{16}$.

Multiprecision Computing Toolbox [28] with the setting `mp.Digits(34)`, which is compliant with the IEEE 754-2008 standard [20].

In each figure in this section, plots on the left show the behavior of the forward error `ferr` (red), normwise relative backward error `nbe` (blue), and componentwise relative backward error `cbe` (green). The dotted black line shows the value of the working precision u. Corresponding plots on the right show bounds on the sizes of the quantities in ϕ_i in the condition (3.9) for convergence of the forward error. Here we plot $2u_s\kappa_\infty(A)\mu_i$ (cyan), $2u_s\,\mathrm{cond}(A)$ (orange), and $u_s\|E_i\|_\infty$ (magenta). The quantity ϕ_i (which is the minimum of the cyan and orange values plus the magenta value) is plotted in black. The dotted black line marks 1. The x-axes are chosen to enable easy comparison between plots for different choices of u_f, u, and u_r.

In describing the choice of precisions we will use notation of the form $(u_f, u, u_r) =$ (half, double, quad), which means that u_f, u, and u_r take the values corresponding to IEEE half, double, and quadruple precisions, respectively.

We begin with an experiment that demonstrates the potential benefits of GMRES-IR (Algorithm 8.1). The working precision is double precision. We generate the matrix A using `gallery('randsvd',1e9,2)`. For this matrix, $\kappa_\infty(A) = 2.0e+10$, and for the linear system $Ax = b$ with the randomly generated b vector, $\mathrm{cond}(A,x) = 5.2e+09$. In Figure 10.1 we show convergence results for iterative refinement with LU factorization using $(u_f, u_r) =$ (single, double) on the first row, (single, quad) on the second row, and (double, quad) on the third row. The fourth row of plots shows results using GMRES-IR with $(u_f, u_r) =$ (single, quad). Above the plots in the last row, we show 'GMRES its', which is a list in which the ith element is the number of iterations that GMRES took to converge in refinement step i (using the convergence criterion that the relative GMRES residual 2-norm is less than or equal to 10^{-4}). From the first row, we see that this system is too ill conditioned to be solved using iterative refinement with a single precision LU factorization; neither the forward error nor the backward errors converge. From the second row, we see that as expected, computing the residuals more precisely has no effect on convergence. The only way to improve convergence using standard iterative refinement is to use higher precision in the LU factorization; we see from the third row that with a double precision LU factorization this results in fast convergence. The fourth row of plots shows the potential gains from using GMRES-IR. Here, even though the LU factors are computed in single precision, the forward errors and backward errors all reach the level of the working precision after two refinement steps, which incur only five GMRES iterations in total.

We now investigate the behavior of both iterative refinement with LU factorization and GMRES-IR (Algorithms 1.1 and 8.1, respectively) in more detail.

10.1. Iterative refinement with LU factorization. We begin by testing iterative refinement (Algorithm 1.1) with LU factorization as the solver, first with two precisions and then three precisions. For each test in this section we list $\kappa_2(A)$ (which we specify as `kappa` when generating the `randsvd` matrix), $\kappa_\infty(A)$, and $\mathrm{cond}(A, x)$ above the corresponding plots.

10.1.1. Iterative refinement in two precisions. When $u_r = u$ and $u_f = u^{1/2}$, so that LU factorization is done at half the working precision, we expect backward errors to converge to level u and forward errors to converge to level $\mathrm{cond}(A, x)u$ for matrices with condition number up to $1/u_f$; see Table 7.1.

Results with $(u_f, u, u_r) = (\text{single}, \text{double}, \text{double})$ are shown in Figure 10.2 for a matrix with condition number well within the limit of $1/u_f$ (top row) and a matrix that is extremely ill conditioned with respect to u_f (bottom row). The observed behavior is consistent with the theory: the forward and backward errors all converge to the expected levels (note the effect of the $\mathrm{cond}(A, x)$ term in the forward error limit). In the second test (bottom row), we see that $\kappa_\infty(A)$ for the generated matrix is already close to $1/u_f$. This causes the convergence factor ϕ_i to be close to 1, and thus many refinement steps are required for convergence. Note from the plots on the right that ϕ_i is dominated by the $u_s\|E_i\|_\infty$ terms.

10.1.2. Iterative refinement in three precisions. We now demonstrate the potential benefits of iterative refinement in three precisions. In Figure 10.3 we take $(u_f, u, u_r) = (\text{single}, \text{double}, \text{quad})$ and use the same matrices as in Figure 10.2. Comparing Figure 10.3 with Figure 10.2 shows the benefit of computing the residuals at twice the working precision: the forward error converges to level u in both cases, without any dependence on the $\mathrm{cond}(A, x)$. Also note that the use of extra precision in the residual computation has no effect on the rate of convergence (compare the values of ϕ_i in the right-hand plots).

10.2. GMRES-based iterative refinement. We now test GMRES-IR (Algorithm 8.1) with the combinations of precisions described in Table 8.1. For these tests, within the GMRES method we use the convergence criterion that the relative residual in the 2-norm is no greater than 10^{-2}, 10^{-4}, and 10^{-6} when u is half precision, single precision, and double precision, respectively. Above the plots for each test, we give $\kappa_2(A)$, $\kappa_\infty(A)$, $\mathrm{cond}(A, x)$, and $\kappa_\infty(\widetilde{A}) = \kappa_\infty(\widehat{U}^{-1}\widehat{L}^{-1}A)$.

10.2.1. GMRES-IR in two precisions. We first test GMRES-IR when two different precisions are used: $u_f = u$ and $u_r = u^2$. This is the special case that was investigated in [9]. Here we expect convergence of the forward and backward errors to level u for matrices with $\kappa_\infty(A)$ up to $1/u$; see Table 8.1.

In Figure 10.4, we use $(u_f, u, u_r) = (\text{half}, \text{half}, \text{single})$ and in Figure 10.5 we use $(u_f, u, u_r) = (\text{single}, \text{single}, \text{double})$. For each combination of precisions, we show results for a matrix with condition number well within the $1/u$ limit (top row) and a matrix that is on the edge of numerical singularity, i.e., $\kappa_\infty(A) \gtrsim 1/u$ (bottom row). For the reasonably well-conditioned matrices, the results are as expected. For the case where $\kappa_\infty(A) \gtrsim 1/u$, the results are better than expected. Despite A being extremely ill conditioned, GMRES-IR succeeds in obtaining backward and forward errors on the level u, and does so requiring very few GMRES iterations in each refinement step.

Notice that with $u_f = u$, $\kappa_\infty(\widetilde{A})$ can be substantially less than $\kappa_\infty(A)$ even when $\kappa_\infty(A)$ is of the order of $1/u$; see, e.g., the bottom row in Figure 10.5.

We note also that the orange pluses in, e.g., Figure 10.5 are at or above 1. That ϕ_i is nevertheless substantially less than 1 is thanks to the min function in (3.9) and the ameliorating effect of μ_i, which was first pointed out in [9].

10.2.2. GMRES-IR in three precisions. We now show the benefit of using three precisions in GMRES-IR, with $u_r < u < u_f$. According to the theory, the forward and backward errors should converge to level u for matrices with $\kappa_\infty(A) \leq 1/u$. In other words, in GMRES-IR we can compute the LU factorization in precision $u^{1/2}$ and still attain the same bounds on the backward and forward errors as if it were computed in precision u.

Tests with (u_f, u, u_r) set to $(\mathrm{half}, \mathrm{single}, \mathrm{double})$ and $(\mathrm{single}, \mathrm{double}, \mathrm{quad})$ are shown in Figures 10.6 and Figure 10.7, respectively. Again for each set of precisions, we show results for a matrix with condition number well within the $1/u$ limit (top rows) and for a matrix which is extremely ill-conditioned with respect to precision u (bottom rows).

The results here are consistent with the theory: in all cases we have convergence of the backward and forward errors to level u. We note that the use of lower precision for the LU factorization can work very well for reasonably well-conditioned matrices. For example, in the top row of Figure 10.7 where $\kappa_\infty(A) = 1.1\mathrm{e}{+}04$, only four total GMRES iterations across two refinement steps are required to obtain the desired forward and backward errors. We note that standard iterative refinement with LU factorization also performs well on this problem (see Figure 10.3).

It is important to point out that the number of GMRES iterations in each refinement step increases significantly with $\kappa_\infty(A)$ for this class of problems. In the bottom rows in both figures, where A is extremely ill-conditioned with respect to u, nearly n GMRES iterations are required for each solve. Since $\widetilde{A}$ is applied in precision u_r in each GMRES iteration, this will not be efficient compared with simply computing the LU factorization more accurately.

To show that this approach can indeed still be efficient for some problems, we now run analogous experiments for problems generated using `randsvd` mode 2, which generates matrices having only one small singular value. The results are shown in Figures 10.8 and 10.9 for GMRES-IR with (u_f, u, u_r) set to $(\mathrm{half}, \mathrm{single}, \mathrm{double})$ and $(\mathrm{single}, \mathrm{double}, \mathrm{quad})$, respectively. For mode 2 matrices, the number of GMRES iterations per refinement step grows more modestly with $\kappa_\infty(A)$. For example, in the bottom row of Figure 10.9, convergence requires only 7 total GMRES iterations even though $\kappa_\infty(A) > 1/u$. Also note here that $\kappa_\infty(\widetilde{A})$ is still very large compared with $\kappa_\infty(A)$, which emphasizes the fact that the GMRES convergence rate cannot be connected with the condition number of the preconditioned matrix.

Finally, we consider the more extreme case of GMRES-IR using precisions $(u_f, u, u_r) = (\mathrm{half}, \mathrm{double}, \mathrm{quad})$. The analysis summarized in Table 8.1 predicts that the forward and backward errors should converge to level $u \approx 10^{-16}$ for matrices with $\kappa_\infty(A)$ up to 10^{12}. In other words, in GMRES-IR we can compute the LU factorization in a quarter of the working precision without increasing the forward or backward errors.

We show tests for `randsvd` mode 3 matrices in Figure 10.10 and `randsvd` mode 2 matrices in Figure 10.11. The story is largely the same as in the case of $(u_f, u, u_r) = (\mathrm{single}, \mathrm{double}, \mathrm{quad})$ in Figures 10.7 and 10.9. For `randsvd` mode 3 matrices, although the errors reach the levels predicted by the theory, each solve may require too

many GMRES iterations to be practical unless A is well conditioned (see Figure 10.10). However, Figure 10.11 shows that for `randsvd` mode 2 matrices the number of GM-RES iterations is much more favorable for ill conditioned A. Note that in the bottom row in Figure 10.10, we encounter overflow in computing the initial solution x_0 and thus take $x_0 = 0$.

10.3. Discussion. The experiments show that the behaviors in practice of Algorithm 1.1 and Algorithm 8.1 (GMRES-IR) match well the predictions of the analysis, and even exceed it for GMRES-IR. An important difference between the two algorithms is that GMRES-IR converges quickly in all cases (at most three iterations in Figures 10.4–10.11), whereas Algorithm 1.1 using LU factorization can be much slower. This is related to the fact, visible in the right-hand columns of the plots, that Algorithm 1.1 with LU factorization is "on the edge" as regards the convergence criteria (ϕ_i is close to 1), whereas GMRES-IR satisfies the criteria much more comfortably.

Our experiments confirm that the LU factorization can be computed at less than the working precision while still obtaining backward errors and forward errors at the working precision.

The overall efficiency of GMRES-IR depends on the number of GMRES iterations required. Using less than working precision in the LU factorization can in some cases diminish the effectiveness of $\widehat{L}\widehat{U}$ as a preconditioner in GMRES-IR, resulting in an undesirably high number of GMRES iterations. This can in turn reduce or outweigh any potential computational savings from computing a lower precision LU factorization.

For ease of comparison between approaches, our experiments use a consistent GM-RES tolerance based on the working precision (10^{-2} for single, 10^{-4} for double, and 10^{-6} for quadruple). In practice, however, the GMRES tolerance could be adjusted to minimize the total number of GMRES iterations performed across refinement steps. The analysis in [9, sec. 3] shows that the smaller $\kappa_\infty(\widetilde{A})$, the larger we can set the GMRES convergence tolerance while still meeting the constraint (2.3); of course, if the tolerance parameter is made too large, this can increase the number of refinement steps required for convergence.

Of course, whether changing the GMRES tolerance will result in fewer GMRES iterations depends on the convergence trajectory of GMRES, which in turn depends heavily on properties of the linear system. For nonnormal matrices, $\kappa_\infty(\widetilde{A})$ and even the full spectrum of $\widetilde{A}$, have no direct connection to the GMRES convergence rate, so we cannot draw any theoretical conclusions.

For normal A, however, we can connect spectral properties of $\widetilde{A}$ with the convergence rate of GMRES. Although further investigation is out of the scope of this work, we briefly mention some potential approaches for improving GMRES-IR performance in cases where the coefficient matrix is normal and GMRES convergence for the resulting preconditioned matrix $\widetilde{A}$ constructed using lower precision LU factors is slow. One possibility is to add an additional preconditioner in order to eliminate eigenvalues or clusters of eigenvalues that cause difficulties for GMRES. One could also incorporate a deflation-based technique to eliminate these parts of the spectrum. Another approach (for any A) is to try a different Krylov subspace iterative method; though it may not have the same guarantees on backward stability, it may provide a faster convergence rate in practice.

11. Conclusions. This work makes two main contributions to the solution of $Ax = b$. The first contribution is to show that by using three precisions instead of two in iterative refinement, it is possible to accelerate the solution process and to obtain more accurate results for a wider class of problems. To be concrete, let the working precision in which A, b, and the iterates x_i are stored be IEEE single precision and consider the following four forms of iterative refinement, all employing LU factorization.

- Method 1 (traditional): factorize A at single precision, compute residuals at double precision.
- Method 2 (Langou et al. [23], with single and double precision therein replaced by half and single precision, respectively): factorize A at half precision, compute residuals at single precision.
- Algorithm 1.1: factorize A at half precision, compute residuals at double precision.
- GMRES-IR (Algorithm 8.1): factorize A at half precision, compute residuals at double precision, compute updates using preconditioned GMRES.

Method 1 is guaranteed to provide forward and backward errors of order $u \approx 10^{-8}$ as long as $\kappa_\infty(A) < 10^8$. Method 2 is potentially up to twice as fast, since it factorizes at half precision, but it requires $\kappa_\infty(A) < 10^4$ to guaranteed convergence and it delivers a forward error of order $\mathrm{cond}(A, x)u$. Algorithm 1.1 improves on Method 2 by delivering a forward error of order u under the same assumption on $\kappa_\infty(A)$. GMRES-IR provides a further improvement because it requires only $\kappa_\infty(A) < 10^8$ for convergence, like Method 1. Moreover, it is likely to converge faster than Method 2 and Algorithm 1.1.

The overall speed of GMRES-IR in three precisions depends on the number of GMRES iterations, which is hard to predict and can be large. However, we have shown experimentally that for some problems GMRES can converge in a small number of iterations. (When GMRES-IR is used with just two precisions, as originally proposed in [9], fast convergence of GMRES is always observed in our experience.)

Further work is needed to tune the GMRES convergence tolerance, to investigate alternative GMRES preconditioning strategies, and to investigate the speed of the algorithms in computing environments where half, single, and double precisions are supported in hardware.

These results can be viewed in a different way by comparison with a standard $Ax = b$ solver based on LU factorization in precision u. By using Algorithm 1.1 or GMRES-IR with $u_f = u^{1/2}$ we can solve the system more accurately and up to twice as fast, the speed advantage arising because the $O(n^3)$ part of the work is potentially all done at lower precision.

The second contribution of this work is to give general backward error and forward error analyses of iterative refinement that include almost all previous ones as special cases and improve on some existing results. Crucially, the analyses include four precisions as parameters, which is necessary in order for them to apply to GMRES-IR. Our numerical experiments confirm the predictions of the theory regarding conditions for convergence and the limiting backward and forward errors of Algorithms 1.1 and 8.1. The analyses should be useful for understanding further algorithmic variants that may be proposed, for example ones based on approximate LU factors (such as those from incomplete factorizations), or on different iterative solvers.
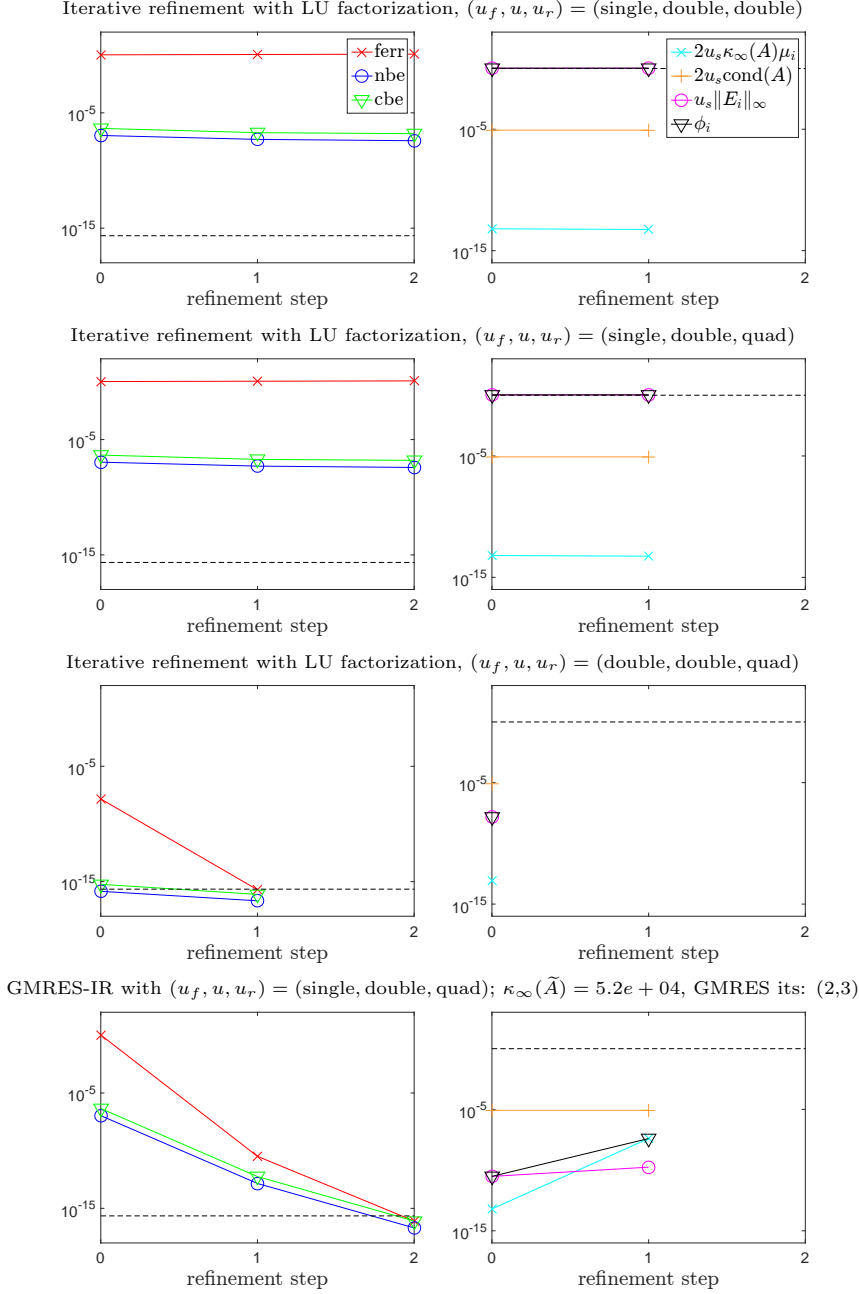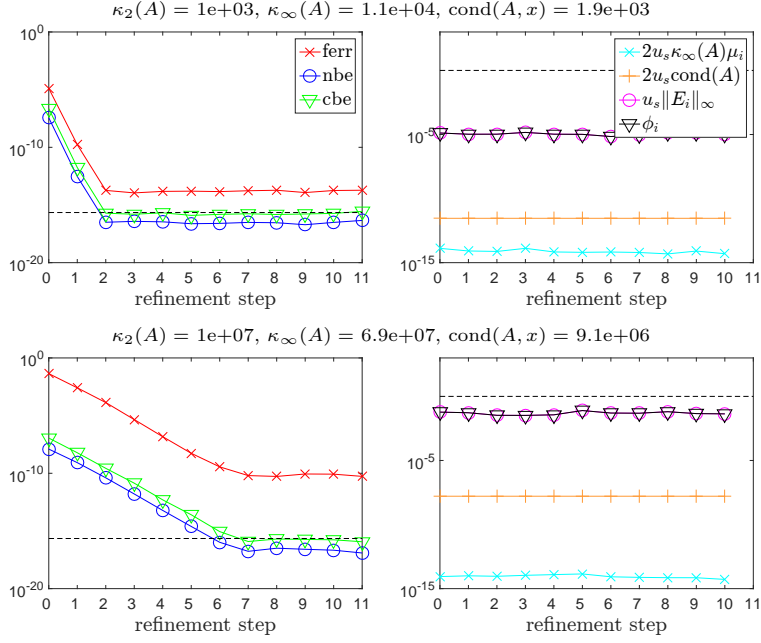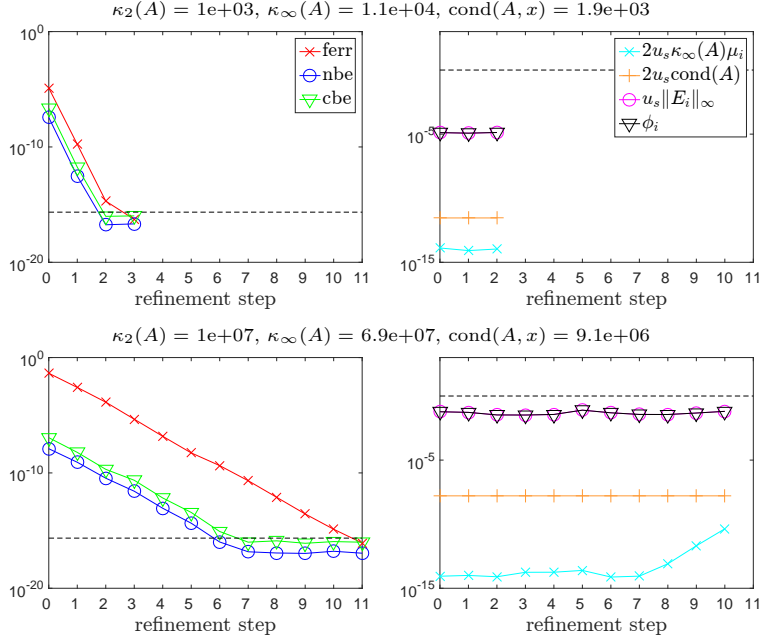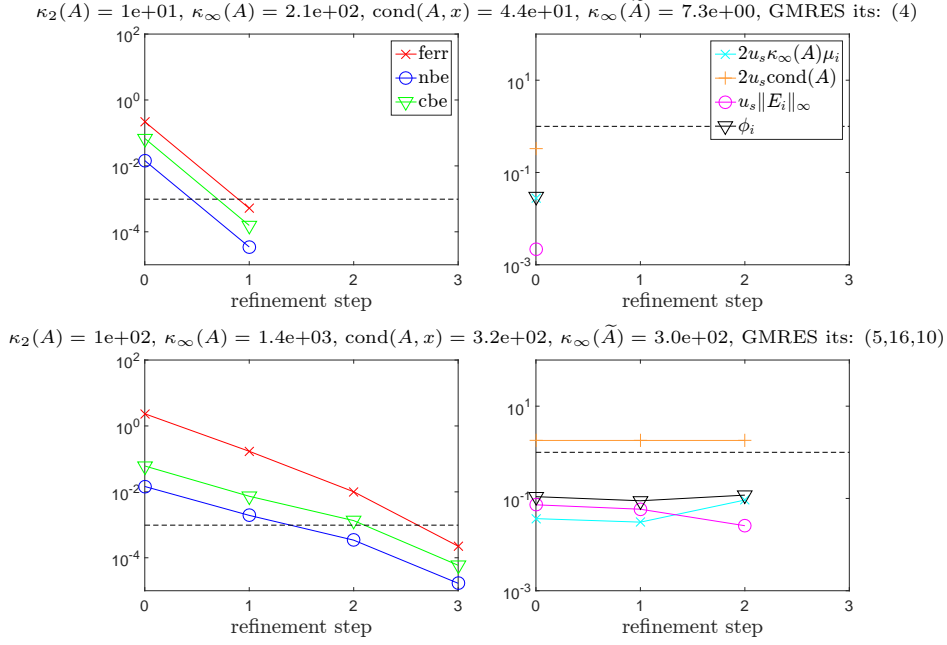
Our MATLAB codes are available at https://github.com/eccarson/ir3.

Iterative refinement with LU factorization, $(u_f, u, u_r) = $ (single, double, double)

Iterative refinement with LU factorization, $(u_f, u, u_r) = $ (single, double, quad)

Iterative refinement with LU factorization, $(u_f, u, u_r) = $ (double, double, quad)

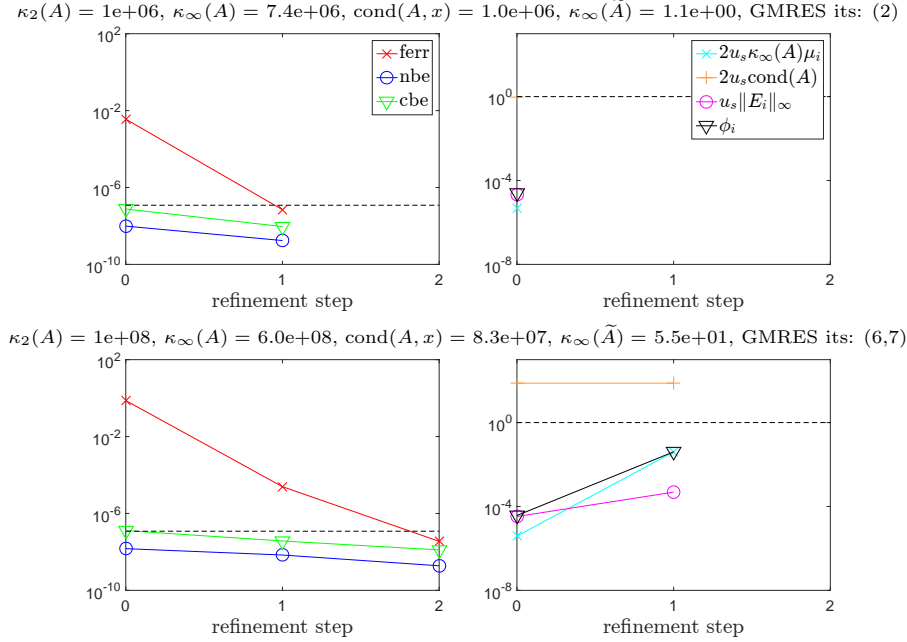GMRES-IR with $(u_f, u, u_r) = $ (single, double, quad); $\kappa_\infty(\widetilde{A}) = 5.2e + 04$, GMRES its: (2,3)
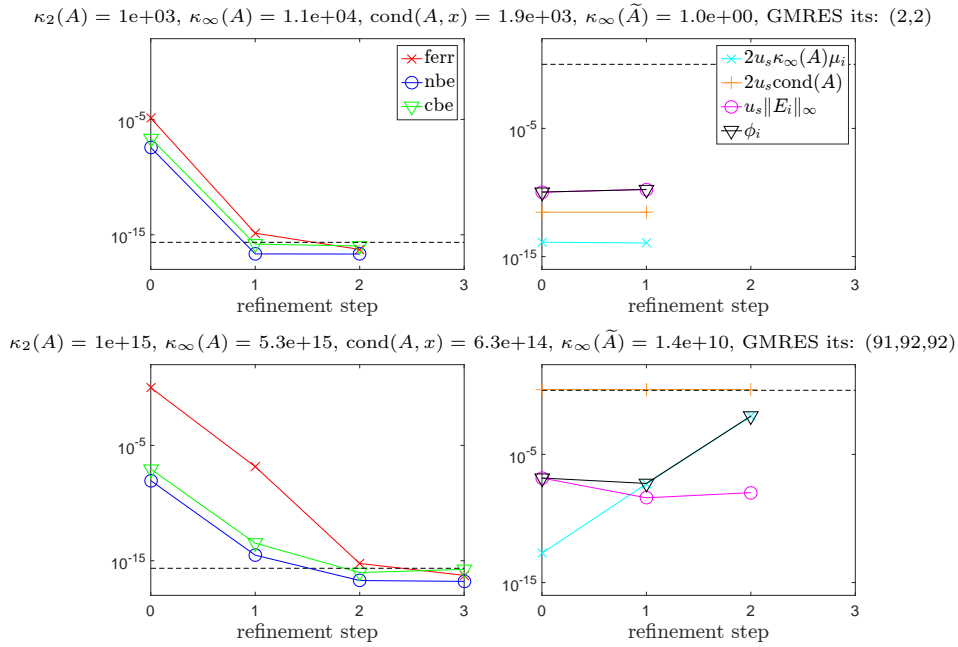

FIG. 10.1. *Comparison of iterative refinement with LU factorization and GMRES-IR for solving $Ax = b$ using various precisions. The matrix is generated using* `gallery('randsvd',100,1e9,2)`. *For this problem, $\kappa_2(A) = 1e + 09$, $\kappa_\infty(A) = 2.0e + 10$, and $\text{cond}(A, x) = 5.2e + 09$.*

FIG. 10.2. *Iterative refinement with LU factorization using* $(u_f, u, u_r) = (\text{single}, \text{double}, \text{double})$.

FIG. 10.3. *Iterative refinement with LU factorization using* $(u_f, u, u_r) = (\text{single}, \text{double}, \text{quad})$.

$\kappa_2(A) = 1e+01$, $\kappa_\infty(A) = 2.1e+02$, $\text{cond}(A, x) = 4.4e+01$, $\kappa_\infty(\widetilde{A}) = 7.3e+00$, GMRES its: (4)

$\kappa_2(A) = 1e+02$, $\kappa_\infty(A) = 1.4e+03$, $\text{cond}(A, x) = 3.2e+02$, $\kappa_\infty(\widetilde{A}) = 3.0e+02$, GMRES its: (5,16,10)

FIG. 10.4. *GMRES-IR using* $(u_f, u, u_r) = (\text{half}, \text{half}, \text{single})$.

$\kappa_2(A) = 1e+06$, $\kappa_\infty(A) = 7.4e+06$, $\text{cond}(A, x) = 1.0e+06$, $\kappa_\infty(\widetilde{A}) = 1.1e+00$, GMRES its: (2)

$\kappa_2(A) = 1e+08$, $\kappa_\infty(A) = 6.0e+08$, $\text{cond}(A, x) = 8.3e+07$, $\kappa_\infty(\widetilde{A}) = 5.5e+01$, GMRES its: (6,7)

FIG. 10.5. *GMRES-IR using* $(u_f, u, u_r) = (\text{single}, \text{single}, \text{double})$.

$\kappa_2(A) = 1\mathrm{e}+01$, $\kappa_\infty(A) = 2.1\mathrm{e}+02$, $\mathrm{cond}(A,x) = 4.4\mathrm{e}+01$, $\kappa_\infty(\widetilde{A}) = 7.7\mathrm{e}+00$, GMRES its: (7,7)

$\kappa_2(A) = 1\mathrm{e}+06$, $\kappa_\infty(A) = 7.4\mathrm{e}+06$, $\mathrm{cond}(A,x) = 1.0\mathrm{e}+06$, $\kappa_\infty(\widetilde{A}) = 5.8\mathrm{e}+05$, GMRES its: (90,100,91)
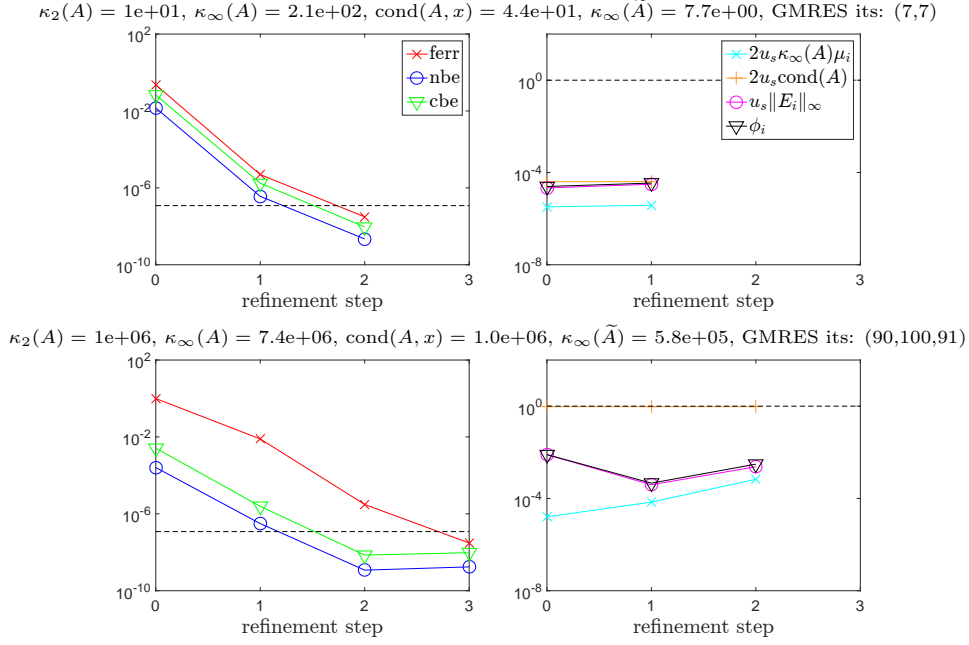
FIG. 10.6. *GMRES-IR using* $(u_f, u, u_r) = (\mathrm{half}, \mathrm{single}, \mathrm{double})$.

$\kappa_2(A) = 1\mathrm{e}+03$, $\kappa_\infty(A) = 1.1\mathrm{e}+04$, $\mathrm{cond}(A,x) = 1.9\mathrm{e}+03$, $\kappa_\infty(\widetilde{A}) = 1.0\mathrm{e}+00$, GMRES its: (2,2)

$\kappa_2(A) = 1\mathrm{e}+15$, $\kappa_\infty(A) = 5.3\mathrm{e}+15$, $\mathrm{cond}(A,x) = 6.3\mathrm{e}+14$, $\kappa_\infty(\widetilde{A}) = 1.4\mathrm{e}+10$, GMRES its: (91,92,92)

FIG. 10.7. *GMRES-IR using* $(u_f, u, u_r) = (\mathrm{single}, \mathrm{double}, \mathrm{quad})$.

$\kappa_2(A) = 1e+01$, $\kappa_\infty(A) = 2.1e+02$, $\text{cond}(A, x) = 5.5e+01$, $\kappa_\infty(\widetilde{A}) = 1.4e+01$, GMRES its: (5,6)
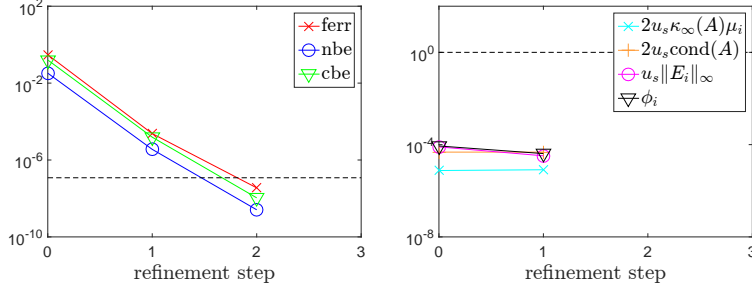
$\kappa_2(A) = 1e+05$, $\kappa_\infty(A) = 2.0e+06$, $\text{cond}(A, x) = 5.2e+05$, $\kappa_\infty(\widetilde{A}) = 7.7e+06$, GMRES its: (7,24,9)
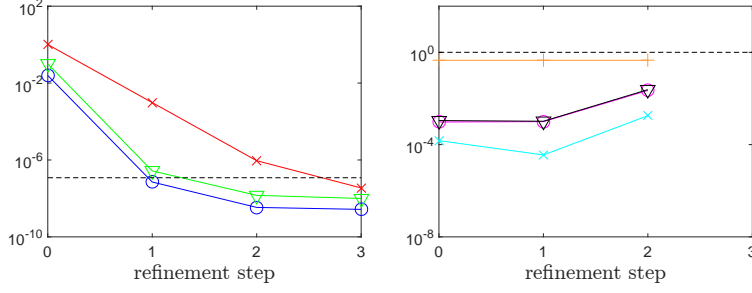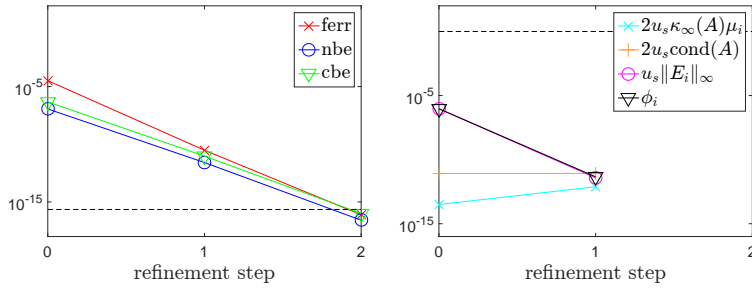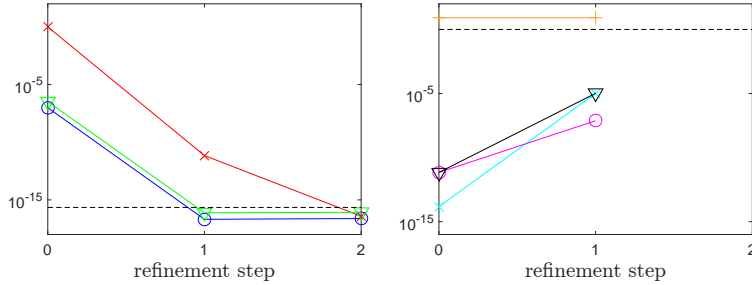

FIG. 10.8. *GMRES-IR using* $(u_f, u, u_r) = (\text{half}, \text{single}, \text{double})$. *The matrices are generated using* `randsvd` *mode 2 (cf. Figure 10.6, which uses mode 3).*

$\kappa_2(A) = 1e+03$, $\kappa_\infty(A) = 2.0e+04$, $\text{cond}(A, x) = 5.2e+03$, $\kappa_\infty(\widetilde{A}) = 1.0e+00$, GMRES its: (1,2)

$\kappa_2(A) = 1e+15$, $\kappa_\infty(A) = 1.9e+16$, $\text{cond}(A, x) = 5.1e+15$, $\kappa_\infty(\widetilde{A}) = 1.6e+10$, GMRES its: (3,4)
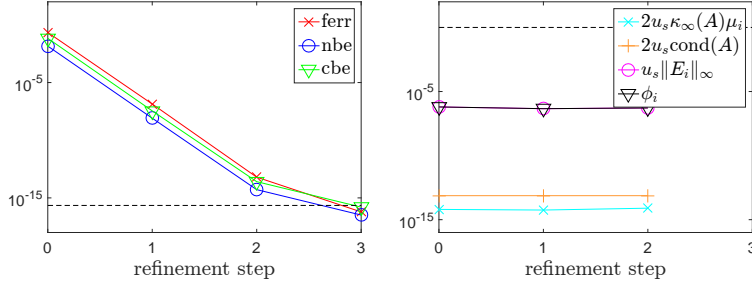

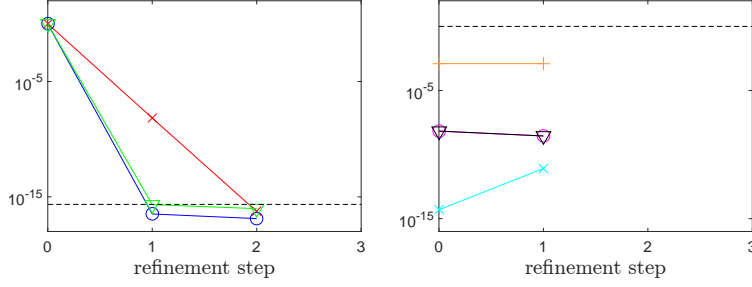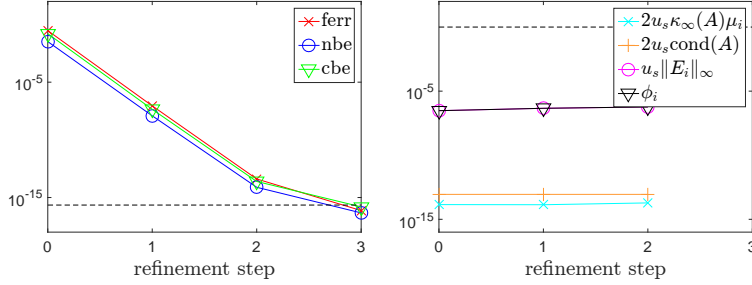FIG. 10.9. *GMRES-IR using* $(u_f, u, u_r) = (\text{single}, \text{double}, \text{quad})$. *The matrices are generated using* `randsvd` *mode 2 (cf. Figure 10.7, which uses mode 3).*

$\kappa_2(A) = 1\text{e}{+}01,\ \kappa_\infty(A) = 2.1\text{e}{+}02,\ \text{cond}(A, x) = 4.4\text{e}{+}01,\ \kappa_\infty(\widetilde{A}) = 7.4\text{e}{+}00,\ \text{GMRES its: } (9,9,9)$

$\kappa_2(A) = 1\text{e}{+}12,\ \kappa_\infty(A) = 5.7\text{e}{+}12,\ \text{cond}(A, x) = 6.8\text{e}{+}11,\ \kappa_\infty(\widetilde{A}) = 6.0\text{e}{+}11,\ \text{GMRES its: } (100,100)$

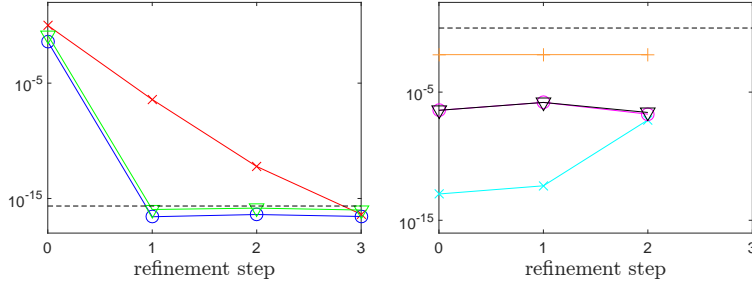FIG. 10.10. *GMRES-IR using* $(u_f, u, u_r) = (\text{half}, \text{double}, \text{quad})$. *The matrices are generated using* `randsvd` *mode 3.*

$\kappa_2(A) = 1\text{e}{+}01,\ \kappa_\infty(A) = 2.1\text{e}{+}02,\ \text{cond}(A, x) = 5.5\text{e}{+}01,\ \kappa_\infty(\widetilde{A}) = 1.4\text{e}{+}01,\ \text{GMRES its: } (8,8,8)$

$\kappa_2(A) = 1\text{e}{+}12,\ \kappa_\infty(A) = 2.0\text{e}{+}13,\ \text{cond}(A, x) = 5.2\text{e}{+}12,\ \kappa_\infty(\widetilde{A}) = 1.4\text{e}{+}13,\ \text{GMRES its: } (17,19,19)$

FIG. 10.11. *GMRES-IR using* $(u_f, u, u_r) = (\text{half}, \text{double}, \text{quad})$. *The matrices are generated using* `randsvd` *mode 2.*

REFERENCES

[1] A. Abdelfattah, H. Anzt, J. Dongarra, M. Gates, A. Haidar, J. Kurzak, P. Luszczek, S. Tomov, I. Yamazaki, and A. YarKhan. Linear algebra software for large-scale accelerated multicore computing. *Acta Numerica*, 25:1–160, 2016.

[2] Emmanuel Agullo, Jim Demmel, Jack Dongarra, Bilel Hadri, Jakub Kurzak, Julien Langou, Hatem Ltaief, Piotr Luszczek, and Stanimire Tomov. Numerical linear algebra on emerging architectures: the PLASMA and MAGMA projects. *Journal of Physics: Conference Series*, 180(1):012037, 2009.

[3] E. Anderson, Z. Bai, C. H. Bischof, S. Blackford, J. W. Demmel, J. J. Dongarra, J. J. Du Croz, A. Greenbaum, S. J. Hammarling, A. McKenney, and D. C. Sorensen. *LAPACK Users' Guide*. Third edition, Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 1999. xxvi+407 pp. ISBN 0-89871-447-8.

[4] Edward Anderson. Robust triangular solves for use in condition estimation. Technical Report CS-91-142, Department of Computer Science, University of Tennessee, Knoxville, TN, USA, August 1991. 35 pp. LAPACK Working Note 36.

[5] M. Arioli, J. W. Demmel, and I. S. Duff. Solving sparse linear systems with sparse backward error. *SIAM J. Matrix Anal. Appl.*, 10(2):165–190, 1989.

[6] M. Arioli and I. S. Duff. Using FGMRES to obtain backward stability in mixed precision. *Electron. Trans. Numer. Anal.*, 33:31–44, 2009.

[7] 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(5):2024–2044, 2007.

[8] M. Arioli and J. Scott. Chebyshev acceleration of iterative refinement. *Numer. Algorithms*, 66 (3):591–608, 2014.

[9] Erin Carson and Nicholas J. Higham. A new analysis of iterative refinement and its application to accurate solution of ill-conditioned sparse linear systems. MIMS EPrint 2017.12, Manchester Institute for Mathematical Sciences, The University of Manchester, UK, March 2017. 23 pp. Revised July 2017. Submitted to SIAM J. Sci. Comput.

[10] James W. Demmel and Xiaoye Li. Faster numerical algorithms via exception handling. *IEEE Trans. Comput.*, 43(8):983–992, 1994.

[11] Craig C. Douglas, Jan Mandel, and Willard L. Miranker. Fast hybrid solution of algebraic systems. *SIAM J. Sci. Statist. Comput.*, 11(6):1073–1086, 1990.

[12] L. Fox, H. D. Huskey, and J. H. Wilkinson. Notes on the solution of algebraic linear simultaneous equations. *Quart. J. Mech. Appl. Math.*, 1:149–173, 1948.

[13] L. Fox, H. D. Huskey, and J. H. Wilkinson. The solution of algebraic linear simultaneous equations by punched card methods. Report, Mathematics Division, Department of Scientific and Industrial Research, National Physical Laboratory, Teddington, UK, 1948. 8 pp. "This note was intended to be included as" section 5 of [12], "but was finally omitted for reasons of economy of space".

[14] Philip E. Gill, Michael A. Saunders, and Joseph R. Shinnerl. On the stability of Cholesky factorization for symmetric quasidefinite systems. *SIAM J. Matrix Anal. Appl.*, 17(1): 35–46, 1996.

[15] Anne Greenbaum, Vlastimil Pták, and Zdeněk Strakoš. Any nonincreasing convergence curve is possible for GMRES. *SIAM J. Matrix Anal. Appl.*, 17(3):465–469, 1996.

[16] Nicholas J. Higham. Iterative refinement enhances the stability of QR factorization methods for solving linear equations. *BIT*, 31:447–468, 1991.

[17] Nicholas J. Higham. Iterative refinement for linear systems and LAPACK. *IMA J. Numer. Anal.*, 17(4):495–509, 1997.

[18] Nicholas J. Higham. *Accuracy and Stability of Numerical Algorithms*. Second edition, Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2002. xxx+680 pp. ISBN 0-89871-521-0.

[19] 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(2):17:1–17:24, 2010.

[20] *IEEE Standard for Floating-Point Arithmetic, IEEE Std 754-2008 (revision of IEEE Std 754-1985)*. IEEE Computer Society, New York, 2008. 58 pp. ISBN 978-0-7381-5752-8.

[21] M. Jankowski and H. Woźniakowski. Iterative refinement implies numerical stability. *BIT*, 17: 303–311, 1977.

[22] Andrzej Kiełbasiński. Iterative refinement for linear systems in variable-precision arithmetic. *BIT*, 21(1):97–103, 1981.

[23] Julie Langou, Julien Langou, Piotr Luszczek, Jakub Kurzak, Alfredo Buttari, and Jack Dongarra. Exploiting the performance of 32 bit floating point arithmetic in obtaining 64 bit

accuracy (revisiting iterative refinement for linear systems). In *Proceedings of the 2006 ACM/IEEE Conference on Supercomputing*, November 2006.

[24] Jörg Liesen and Petr Tichỳ. The worst-case GMRES for normal matrices. *BIT*, 44(1):79–98, 2004.

[25] Cleve B. Moler. Cleve Laboratory. `http://uk.mathworks.com/matlabcentral/fileexchange/59085-cleve-laborator`.

[26] Cleve B. Moler. Iterative refinement in floating point. *J. Assoc. Comput. Mach.*, 14(2):316–321, 1967.

[27] Cleve B. Moler. "Half precision" 16-bit floating point arithmetic. `http://blogs.mathworks.com/cleve/2017/05/08/half-precision-16-bit-floating-point-arithmetic/`, May 2017.

[28] Multiprecision Computing Toolbox. Advanpix, Tokyo. `http://www.advanpix.com`.

[29] W. Oettli and W. Prager. Compatibility of approximate solution of linear equations with given error bounds for coefficients and right-hand sides. *Numer. Math.*, 6:405–409, 1964.

[30] J. L. Rigal and J. Gaches. On the compatibility of a given solution with the data of a linear system. *J. Assoc. Comput. Mach.*, 14(3):543–548, 1967.

[31] Robert D. Skeel. Iterative refinement implies numerical stability for Gaussian elimination. *Math. Comp.*, 35(151):817–832, 1980.

[32] Alicja Smoktunowicz and Jolanta Sokolnicka. Binary cascades iterative refinement in doubled-mantissa arithmetics. *BIT*, 24(1):123–127, 1984.

[33] G. W. Stewart. *Introduction to Matrix Computations*. Academic Press, New York, 1973. xiii+441 pp. ISBN 0-12-670350-7.

[34] Françoise Tisseur. Newton's method in floating point arithmetic and iterative refinement of generalized eigenvalue problems. *SIAM J. Matrix Anal. Appl.*, 22(4):1038–1057, 2001.

[35] J. H. Wilkinson. Progress report on the Automatic Computing Engine. Report MA/17/1024, Mathematics Division, Department of Scientific and Industrial Research, National Physical Laboratory, Teddington, UK, April 1948. 127 pp.

[36] J. H. Wilkinson. *Rounding Errors in Algebraic Processes*. Notes on Applied Science No. 32, Her Majesty's Stationery Office, London, 1963. vi+161 pp. Also published by Prentice-Hall, Englewood Cliffs, NJ, USA. Reprinted by Dover, New York, 1994. ISBN 0-486-67999-3.

[37] J. H. Wilkinson. Modern error analysis. *SIAM Rev.*, 13(4):548–568, 1971.

[38] Zahari Zlatev. Use of iterative refinement in the solution of sparse linear systems. *SIAM J. Numer. Anal.*, 19(2):381–399, 1982.