Electronic Transactions on Numerical Analysis.
Volume 33, pp. 31-44, 2009.

ISSN 1068-9613.

ETNA
Kent State University
etna@mcs.kent.edu

USING FGMRES TO OBTAIN BACKWARD STABILITY IN MIXED PRECISION*

M. ARIOLI† AND I. S. DUFF†

Dedicated to Gérard Meurant on the occasion of his 60th birthday

Abstract. We consider the triangular factorization of matrices in single-precision arithmetic and show how these factors can be used to obtain a backward stable solution. Our aim is to obtain double-precision accuracy even when the system is ill-conditioned. We examine the use of iterative refinement and show by example that it may not converge. We then show both theoretically and practically that the use of FGMRES will give us the result that we desire with fairly mild conditions on the matrix and the direct factorization. We perform extensive experiments on dense matrices using MATLAB and indicate how our work extends to sparse matrix factorization and solution.

Key words. FGMRES, mixed precision arithmetic, hybrid method, direct factorization, iterative methods, large sparse systems, error analysis

AMS subject classifications. 65F05, 65F10, 65F50, 65G20, 65G50

1. Introduction.

We are concerned with the solution of

$$Ax = b, \tag{1.1}$$

when A is an $n \times n$ matrix and x and b are vectors of length n. For most of our discussion the matrix A is dense and unsymmetric, although we will consider the case of sparse symmetric A in Section 6. We will solve these systems using a direct method where the matrix A is first factorized as

$$A \to LU,$$

where L and U are triangular matrices. The solution is then obtained through forward elimination

$$Ly = b$$

followed by back substitution

$$Ux = y,$$

where we have omitted permutations required for numerical stability and sparsity preservation for the sake of clarity. When A is symmetric, we use an LDL^T factorization where the matrix D is block diagonal with blocks of order 1 and 2, so that we can stably factorize indefinite systems.

On many emerging computer architectures, single-precision arithmetic (by which we mean working with 32-bit floating-point numbers) is faster than double-precision arithmetic. In fact on the Cell processor, using single precision can be more than ten times as fast as using double precision [4]. In addition, single-precision numbers require half the storage of double-precision numbers, and the movement of data between memory, cache, and processing units is much reduced by using single rather than double precision. However, in many applications,

*Received February 6, 2008. Accepted August 25, 2008. Published online on January 7, 2009. Recommended by Yousef Saad.

†STFC Rutherford Appleton Laboratory, Chilton, Didcot, Oxfordshire, OX11 0QX, UK. email: mario.arioli@stfc.ac.uk, iain.duff@stfc.ac.uk. Phone: +441235 445332. This work was supported by the EPSRC Grants EP/E053351/1 and EP/F006535/1.

ETNA
Kent State University
etna@mcs.kent.edu

a higher accuracy is required than single precision (with a value of machine precision around 10^{-7}), because the matrix can be so ill-conditioned that single-precision calculation is unable to obtain accuracy to even one significant figure — that is, the results are meaningless.

The loss of accuracy due to ill-conditioning could be even more dramatic if we want to achieve extended-precision accuracy in the residual when the LU factorization is performed in double precision [7]. In some applications it is well known that the matrix will be very ill-conditioned, but a reasonable solution is still required. Because extended precision is typically implemented in software, an extended-precision LU factorization will be prohibitively slow.

In this paper, we show how to use selective double-precision post-processing to obtain solutions with a backward error (scaled residual) of double-precision accuracy (machine precision $\varepsilon \approx 10^{-16}$) even when the factorization is computed in single precision. We show that iterative refinement in double precision may fail when the matrix is ill-conditioned, and then show that, even for such badly-behaved matrices, the use of FGMRES [15] can produce answers to the desired level of accuracy; that is, the solution process using FGMRES is backward stable at the level of double precision. We prove that, under realistic assumptions on the matrix and the factorization, a double-precision FGMRES iteration preconditioned with a single-precision LU factorization is backward stable. Buttari et al. [4] have performed extensive performance testing of similar algorithms on a range of modern architectures and illustrate well that such an approach can be beneficial, but they have no analysis of the algorithms. For the case when A is sparse, which we discuss in Section 6, extensive numerical experiments have been performed by Buttari et al. [5] and Hogg and Scott [10], although again neither paper has any analysis. This paper expands on existing theory of the authors and others to provide a rigorous theoretical background for such mixed arithmetic computations.

We observe that our analysis is still valid if we want to achieve extended-precision accuracy in the residual when the LU factorization is performed in double precision.

We briefly discuss iterative refinement and FGMRES in Section 2 and prove the convergence of the mixed-precision FGMRES algorithm in Section 3. We describe our construction of dense test matrices in Section 4 and illustrate the performance of our algorithms in Section 5. We then show how this can be extended to sparse matrices in Section 6 where we perform the single-precision factorization using the HSL code `MA57` [6]. We present some conclusions in Section 7.

2. Iterative refinement and FGMRES. The standard technique for improving a solution to (1.1) is iterative refinement. This consists of computing the residual

$$r^{(k)} = b - Ax^{(k)} \tag{2.1}$$

to the current approximate solution $x^{(k)}$ and calculating a correction to $x^{(k)}$ by solving

$$A\,\delta x^{(k)} = r^{(k)}. \tag{2.2}$$

The new estimate is then

$$x^{(k+1)} = x^{(k)} + \delta x^{(k)}.$$

The solution of the correction equation (2.2) will of course use the original factorization of A, and so can be performed relatively quickly. It is easy to see that the condition for the convergence of iterative refinement is that the spectral radius of $I - MA$ is less than one, where M is the approximation to A^{-1} obtained using the factorization of A.

ETNA
Kent State University
etna@mcs.kent.edu

Originally, it was customary to compute the residual in higher precision [18]. More recently, Skeel [17] established that in order to reduce the scaled residual (backward error) to machine precision, it is only necessary to compute the residual and correction in the same precision as the original computation. However, since we wish to obtain solutions with double-precision accuracy when using a single-precision factorization, we will follow the original recommendation and compute the residuals in double precision.

One potentially major restriction when using iterative refinement is the condition on the spectral radius of $I - MA$. If M is not a very accurate factorization for A then this condition may not be met. We have discussed [1] at length the case when A is sparse and the factorization is computed using static pivoting, for example using the HSL code `MA57` [6]. There we have shown that in cases when iterative refinement fails, FGMRES [15] will normally work and is far more robust than either iterative refinement or GMRES [16].

In this current work, we also compute an M which is potentially far from A^{-1} because we compute it in single precision. This will be particularly the case when the condition number of the matrix is large, say around the inverse of single-precision rounding (10^7). We study the use of FGMRES in this context both experimentally (Section 5) and theoretically (Section 3).

The FGMRES algorithm is an Arnoldi method based on Krylov sequences, and we present its restarted variant in detail as Algorithm 2.1. The main reason why FGMRES is superior to GMRES is that FGMRES computes and stores a second set of vectors Z_k corresponding to the preconditioned problem along with the usual orthonormal sequence v_k [1].

ALGORITHM 2.1.

procedure [x] = FGMRES(A,M_i,b,maxit)
$x_0 = M_0^{-1}b$, $r_0 = b - Ax_0$ *and* $\beta = \|r_0\|$
$v_1 = r_0/\beta$; *k = 0; it = 0; convergence = false;*
while convergence = false and $it < maxit$
$k = k + 1$; $it = it + 1$;
$z_k = M_k^{-1}v_k$; $w = Az_k$;
for $i = 1, \ldots, k$ *do*
$h_{i,k} = v_i^T w$;
$w = w - h_{i,k}v_i$;
end for;
$h_{k+1,k} = \|w\|$;
$Z_k = [z_1, \ldots, z_k]$; $H_k = \{h_{i,j}\}_{1 \le i \le j+1; 1 \le j \le k}$;
$y_k = \arg\min_y \|\beta e_1 - H_k y\|$;
if $\|\beta e_1 - H_k y_k\| \le \varepsilon\,(\|b\| + \|A\|\,\|x_k\|)$ *do*
$x_k = x_0 + Z_k y_k$ *and* $r = b - Ax_k$;
if $\|r\| > \varepsilon\,(\|b\| + \|A\|\,\|x_k\|)$ *do*
$x_0 = x_k$, $r_0 = b - Ax_k$, *and* $\beta = \|r_0\|$;
$v_1 = r_0/\beta$; *k = 0; convergence = false;*
else
convergence = true ;
end if
else
$v_{k+1} = w/h_{k+1,k}$; $V_{k+1} = [v_1, \ldots, v_{k+1}]$;
end if
end while;
end procedure.

ETNA
Kent State University
etna@mcs.kent.edu

3. Theoretical proof of convergence for FGMRES. We present here an improved version of Theorem 5.1 in [1] using componentwise bounds for the matrix by vector products. This simplifies the proof given in [1] and gives more accurate bounds on the norm of the residual. In Section 3.1, we discuss the restarting process and we indicate how we use it to achieve normwise backward stability. The first part of our analysis is independent of the choice of the matrices M_i in FGMRES (Algorithm 2.1). The only thing that we assume is that the computed version of the Z_k matrix

$$Z_k = [z_1, \ldots, z_k]$$

is of rank k for all values of $k \leq n$. This guarantees convergence of the algorithm. We later show how mixed precision influences the rank of Z_k and the convergence of the algorithm. Furthermore, the roundoff error analysis of FGMRES in Theorem 3.1 is independent of the specific choice of the computed $\bar{z}_k$ at each step if the resulting computed $\bar{Z}_k$ is full rank.

Under this assumption and following the discussion in [1], we decompose Algorithm 2.1 into three main sub-algorithms:

- Computation of the matrices $C^{(k)}$, V_k, and R_k by the Modified Gram-Schmidt algorithm (MGS) such that
$$C^{(k)} = [r_0, AZ_k] = V_{k+1}R_k; \qquad V_j^T V_j = I_j \quad \forall j, \tag{3.1}$$
where
$$R_k = \begin{bmatrix} \beta e_1 & H_k \end{bmatrix}, \tag{3.2}$$
$$AZ_k = V_{k+1}H_k, \tag{3.3}$$
and H_k is upper Hessenberg. Column $k+1$ of $C^{(k+1)}$ is computed after the kth step of MGS in (3.1) and (3.2) by computing or choosing a new $\bar{z}_{k+1}$. We then generate the next column of V_{k+2} and R_{k+1}.
- Computation of the vector y_k by solving the least-squares problem
$$\min_y \|\beta e_1 - H_k y\| \tag{3.4}$$
using a QR algorithm based on Givens rotations and the upper Hessenberg structure of H_k.
- Computation of $x_k = x_0 + Z_k y_k$ when the residual $\|\beta e_1 - H_k y_k\|$ is less than or equal to the prescribed threshold.

In the following, we denote by $c_p(n, j)$ functions that depend only on the dimension n and the integer j. If the second index is omitted then the function depends only on n. We avoid a precise formulation of these dependences, but we assume that each $c_p(n, j)$ grows moderately with n and j. Finally, if $B \in \mathbb{R}^{p \times q}, p \geq q$ is a full rank matrix, we denote by $\kappa(B) = \|B\| \, \|B^\dagger\|$ its spectral condition number, where $B^\dagger = (B^T B)^{-1} B$. For all matrices and vectors we denote by $|B|$ the matrix or vector of the absolute values. Furthermore, we denote the computed quantities of R_k, V_k, H_k, y_k, r_k, and x_k by the same symbol with a bar above it, i.e., the matrix $\bar{H}_k$ will be the computed value of the matrix H_k.

THEOREM 3.1. *If we apply Algorithm 2.1 to solve (1.1), using finite-precision arithmetic conforming to IEEE standard with relative precision ε and under the following hypotheses:*

$$2.12(n+1)\varepsilon < 0.01 \qquad \textit{and} \qquad c_0(n)\varepsilon\, \kappa(C^{(k)}) < 0.1 \;\forall k, \tag{3.5}$$

where

$$c_0(n) = 18.53 n^{\frac{3}{2}}$$

and

$$|\bar{s}_k| < 1 - \varepsilon\,,\ \forall k, \tag{3.6}$$

where $\bar{s}_k$ *are the sines computed during the Givens algorithm applied to* $\bar{H}_k$ *in order to compute* $\bar{y}_k$*, then there exists* $\hat{k}$*,* $\hat{k} \le n$*, such that,* $\forall k \ge \hat{k}$*, we have*

$$\begin{aligned} \|b - A\bar{x}_k\| \quad \le \quad & c_1(n,k)\varepsilon \Big(\|b\| + \|A\|\,\|\bar{x}_0\| + \\ & \|A\|\,\|\,|\bar{Z}_k|\,|\bar{y}_k|\,\| + \|A\bar{Z}_k\|\,\|\bar{y}_k\| \Big) + \mathcal{O}(\varepsilon^{\,2}). \end{aligned} \tag{3.7}$$

The proof is based on the following two lemmata. Note that we will use some componentwise bounds to obtain the factor $\|\,|\bar{Z}_k|\,|\bar{y}_k|\,\|$ in the bound (3.7). In our earlier analysis we had replaced this with the majorizing quantity $\|\bar{Z}_k\|\,\|\bar{y}_k\|$ but found that this was too loose a bound and that the quantity $\|\bar{Z}_k\|\,\|\bar{y}_k\|$ could be very large in our numerical experiments.

LEMMA 3.2. *If we apply MGS to factorize* $C^{(k)}$ *in (3.1), using finite-precision arithmetic conforming to IEEE standard with relative precision* ε *and under the hypotheses (3.5), then there exist orthonormal matrices* $\hat{V}_k$ *such that*

$$\bar{C}^{(k)} = [r_0 + f, A\bar{Z}_k + E_k] = \hat{V}_{k+1}\bar{R}_k \qquad \forall k \le n \tag{3.8}$$

with

$$\begin{aligned} &\|f\| \le c_2(n,1)\varepsilon\,(\|b\| + \|A\|\,\|\bar{x}_0\|) + \mathcal{O}(\varepsilon^{\,2}) \qquad \textit{and} \\ &|E_k| \le c_3(n,k)\varepsilon\,(\|A\bar{Z}_k\|\mathbf{u}_n\mathbf{u}_k^T + |A|\,|\bar{Z}_k|) + \mathcal{O}(\varepsilon^{\,2}), \end{aligned} \tag{3.9}$$

where we denote by $\mathbf{u}_j$ *the vectors of order* j *with all entries equal to* 1*. Moreover, the computed value of* $\bar{V}_k$ *satisfies the relation*

$$\|\bar{V}_k^{+}\| \le 1.3. \tag{3.10}$$

Proof. By standard techniques [9], the computed matrix by vector products and the initial residual $\bar{r}_0$ satisfy the relations

$$\bar{r}_0 = r_0 + f_1 \qquad \|f_1\| \le c_4(n,1)\varepsilon\,(\|b\| + \|A\|\,\|\bar{x}_0\|) + \mathcal{O}(\varepsilon^{\,2}), \tag{3.11}$$

$$\mathrm{fl}(A\bar{Z}_k) = A\bar{Z}_k + F_k^{(1)}, \qquad |F_k^{(1)}| \le c_5(n,k)\varepsilon\,|A|\,|\bar{Z}_k| + \mathcal{O}(\varepsilon^{\,2}). \tag{3.12}$$

Following [3] and [8], the Gram-Schmidt orthogonalization process applied to $\mathrm{fl}(C^{(k)})$ computes an upper triangular matrix $\bar{R}_k$ for which there exists an orthonormal matrix $\hat{V}_{k+1}$ that satisfies the relations:

$$\left\{ \begin{array}{ll} [\bar{r}_0; \mathrm{fl}(A\bar{Z}_k)] + [f_2; F_k^{(2)}] = \hat{V}_{k+1}\bar{R}_k, & \hat{V}_{k+1}^T\hat{V}_{k+1} = I_{k+1} \\ \|f_2\| \le c_6(n,1)\varepsilon\,\|r_0\| + \mathcal{O}(\varepsilon^{\,2}) & \|F_k^{(2)}\| \le c_7(n,k)\varepsilon\,\|A\bar{Z}_k\| + \mathcal{O}(\varepsilon^{\,2}) \end{array} \right. \tag{3.13}$$

under the hypothesis (3.5).

By combining (3.11), (3.12), and (3.13), we have

$$\left\{ \begin{array}{l} [r_0; A\bar{Z}_k] + [f_1 + f_2; F_k^{(1)} + F_k^{(2)}] = \hat{V}_{k+1}\bar{R}_k, \\ \|f_1 + f_2\| = \|f\| \le c_2(n,1)\varepsilon\,(\|b\| + \|A\|\,\|\bar{x}_0\|) + \mathcal{O}(\varepsilon^{\,2}) \quad \text{and} \\ |F_k^{(1)} + F_k^{(2)}| = |E_k| \le c_3(n,k)\varepsilon\,(\|A\bar{Z}_k\|\mathbf{u}_n\mathbf{u}_k^T + |A|\,|\bar{Z}_k|) + \mathcal{O}(\varepsilon^{\,2}). \end{array} \right. \tag{3.14}$$

ETNA
Kent State University
etna@mcs.kent.edu

We note that the third bound in (3.14) is a componentwise bound. This is in contrast to the normwise bound of [1] and gives us the final tighter bound in our main result (3.7).

Finally, a proof of relation (3.10) can be found in [8, 14]. □

LEMMA 3.3. *Applying the QR factorization with Givens rotations to solve*

$$\min_y \|\bar{\beta}e_1 - \bar{H}_k y\| \, , \tag{3.15}$$

using finite-precision arithmetic conforming to IEEE standard with relative precision ε and under the condition

$$0.1 > c_0(n)\varepsilon\,\kappa(\bar{H}_k) + \mathcal{O}(\varepsilon^{\,2}) \quad \forall k, \tag{3.16}$$

there exist an orthonormal matrix $\hat{G}^{[k]}$, a vector $g^{[k]}$, and an upper Hessenberg matrix ΔH such that the computed value $\bar{y}_k$ satisfies the following relations

$$\left\{ \begin{array}{l} \bar{y}_k = \arg\min_y \|\hat{G}^{[k]}\big(\bar{\beta}e_1 + g^{[k]} - (\bar{H}_k + \Delta H_k)y\big)\|, \\ \|\Delta H_k\| \le c_8(k,1)\varepsilon\,\|\bar{H}_k\| + \mathcal{O}(\varepsilon^{\,2}) \textit{ and } \|g^{[k]}\| \le c_9(k,1)\varepsilon\,\bar{\beta} + \mathcal{O}(\varepsilon^{\,2}). \end{array} \right. \tag{3.17}$$

Moreover, the residuals

$$\alpha_k = \|\bar{\beta}e_1 + g^{[k]} - (\bar{H}_k + \Delta H_k)\bar{y}_k\|,$$

satisfy the equations

$$\left\{ \begin{array}{l} \alpha_k = \bar{\beta}\Big(\prod_{j=0}^k |\bar{s}_j|\Big)\Big(\prod_{j=0}^k (1+\zeta_j)\Big) \\ |\zeta_j| \le \varepsilon \quad \forall\, j. \end{array} \right. \tag{3.18}$$

Under hypothesis (3.6), we have that α_k is strictly decreasing to zero and $\alpha_{\hat{k}} = 0$ for some value of $\hat{k} \le n$.

Proof. See [2]. □

We point out that hypothesis (3.5) implies hypothesis (3.16).

The proof of Theorem 3.1, and in particular the proof of inequality (3.7), follows the proof presented in Appendix A of [1], where we take into account the new bounds (3.14) and that the value $\bar{x}_k$ satisfies the relations

$$\left\{ \begin{array}{l} \bar{x}_k = \bar{x}_0 + \bar{Z}_k\bar{y}_k + \delta x_k, \\ |\delta x_k| \le c_{10}(k,1)\varepsilon\; |\bar{Z}_k|\,|\bar{y}_k| \; + \varepsilon\,|\bar{x}_0| + \mathcal{O}(\varepsilon^{\,2}). \end{array} \right. \tag{3.19}$$

REMARK 3.4. *Hypothesis (3.6) can be removed if the IEEE standard (in particular the IEEE-754 standard) for binary floating-point arithmetic is correctly implemented on the computer [13, page 3]. In this case, formula (3.18) can be modified as follows*

$$\left\{ \begin{array}{l} \alpha_k = \bar{\beta}\Big(\prod_{j=0}^k |\bar{s}_j|\Big)\Big(\prod_{j=0}^k (1+\zeta_j)\Big) \\ |\zeta_j| \le \varepsilon \quad \forall\, j \\ |\bar{s}_j| = 1 \rightarrow |\zeta_j| = 0 \\ |\bar{s}_j| \le 1 \quad \forall\, j. \end{array} \right.$$

ETNA
Kent State University
etna@mcs.kent.edu

3.1. Computing $\bar{Z}_k$ using single-precision arithmetic. In this subsection, we choose $M_j = M = \bar{L}\bar{U}$ for all j in Algorithm 2.1, where $\bar{L}$ and $\bar{U}$ are computed by using an LU factorization of A based on IEEE single-precision arithmetic.

We justify the satisfactory convergence behavior of FGMRES when $\bar{z}_j$ the jth column of $\bar{Z}_k$ is computed by solving the system

$$M z_j = v_j. \tag{3.20}$$

The computed solution $\bar{z}_j$ satisfies the relations [9]

$$M\bar{z}_j = \bar{v}_j + w_j \qquad |w_j| \leq f(\varepsilon)c_{11}(n)\, |\bar{L}|\, |\bar{U}|\, |\bar{z}_j|\, . \tag{3.21}$$

If we use single precision during the backward and forward substitution algorithms $f(\varepsilon) \approx \sqrt{\varepsilon}$, otherwise if double precision is used $f(\varepsilon) \approx \varepsilon$. Thus, we have the following relations

$$\begin{aligned} &M\bar{Z}_k = \bar{V}_k + W_k \\ &W_k = [w_1, \ldots, w_k] \\ &|W_k| \leq f(\varepsilon)c_{12}(n)\, |\bar{L}|\, |\bar{U}|\, |\bar{Z}_k|. \end{aligned} \tag{3.22}$$

Multiplying the first equation in (3.19) by M, we have

$$M(\bar{x}_k - \bar{x}_0 - \delta x_k) = M\bar{Z}_k\bar{y}_k, \tag{3.23}$$

and then from (3.22) it follows that

$$M(\bar{x}_k - \bar{x}_0 - \delta x_k) = \bar{V}_k\bar{y}_k + W_k\bar{y}_k. \tag{3.24}$$

Under the hypotheses of Lemma 3.2, $\bar{V}_k^T\bar{V}_k$ is invertible, and thus we obtain

$$\bar{V}_k^+\Big[M(\bar{x}_k - \bar{x}_0 - \delta x_k) - W_k\bar{y}_k\Big] = \bar{y}_k. \tag{3.25}$$

Finally, combining (3.25) with (3.10), (3.19), and (3.22), we have

$$\begin{aligned} |\bar{y}_k| \leq \quad & |\bar{V}_k^+|\Big[|M(\bar{x}_k - \bar{x}_0)| + \varepsilon\,|M|\,|\bar{x}_0| + \\ & c_{13}(n)\, f(\varepsilon)\left(|M| + |\bar{L}|\,|\bar{U}|\right)|\bar{Z}_k|\,|\bar{y}_k|\Big] + \mathcal{O}(\varepsilon^2) \end{aligned} \tag{3.26}$$

and

$$\begin{aligned} \|\bar{y}_k\| \leq \quad & 1.3\Big[\|M(\bar{x}_k - \bar{x}_0)\| + \varepsilon\,\|M\|\,\|\bar{x}_0\| + \\ & c_{14}(n)\, f(\varepsilon)\left(\|M\| + \|\,|\bar{L}|\,|\bar{U}|\,\|\right)\|\,|\bar{Z}_k|\,|\bar{y}_k|\,\|\Big] + \mathcal{O}(\varepsilon^2). \end{aligned} \tag{3.27}$$

Taking into account that $\bar{x}_0$ is computed in Algorithm 2.1 using the single-precision Gaussian factorization of A, we have

$$\|M\bar{x}_0 - b\| \leq f(\varepsilon)n\,\|A\|\,\|\bar{x}_0\|\,\Gamma\,,$$

where, given the computed $\bar{L}$ and $\bar{U}$,

$$\Gamma = \frac{\|\,|\bar{L}|\,|\bar{U}|\|}{\|A\|}.$$

ETNA
Kent State University
etna@mcs.kent.edu

Then we can further simplify the right-hand side of (3.27), taking into account that

$$\|A - M\| \le n\sqrt{\varepsilon}\,\Gamma\|A\|,$$

so that we have

$$\begin{aligned}\|\bar{y}_k\| \le \quad & 1.3\Big[\|A\bar{x}_k - b\| + c_{15}(n)\,(f(\varepsilon\,) + \sqrt{\varepsilon}\,)\;\Gamma\,\|A\|\,\big(\,\|\bar{x}_0\| + \|\bar{x}_k\|\,\big) + \\ & \qquad c_{16}(n)\,f(\varepsilon\,)\,\Gamma\,\|A\|\,\|\bar{Z}_k\|\,\|\bar{y}_k\|\Big] + \mathcal{O}(\varepsilon^{\,2}).\end{aligned} \tag{3.28}$$

Moreover, if

$$\rho = 1.3c_{16}(n)\,f(\varepsilon\,)\,\Gamma\,\|A\|\,\|\bar{Z}_k\| < 1, \tag{3.29}$$

then we have

$$\begin{aligned}\|\bar{y}_k\| \le \quad & \tfrac{1.3}{1-\rho}\Big[\|A\bar{x}_k - b\| + \\ & \qquad (f(\varepsilon\,) + \sqrt{\varepsilon}\,)\;c_{15}(n)\,\Gamma\,\|A\|\,\big(\,\|\bar{x}_0\| + \|\bar{x}_k\|\,\big)\Big] + \mathcal{O}(\varepsilon^{\,2}).\end{aligned} \tag{3.30}$$

Substituting the upper bound of (3.30) for $\|\bar{y}_k\|$ in (3.7) and assuming that

$$\chi = \frac{1.3\,c_{17}(n)}{1-\rho}\,\varepsilon\;\|A\|\,\|\bar{Z}_k\| < 1, \tag{3.31}$$

we have the final bound

$$\begin{aligned}\|b - A\bar{x}_k\| \le \quad & \frac{\varepsilon\;c_{18}(n)}{1-\chi}\Big[\|b\| + \|A\|\,\big(\,\|\bar{x}_0\| + \|\bar{x}_k\|\,\big)\times \\ & \qquad \Big(1 + (\,f(\varepsilon\,) + \sqrt{\varepsilon}\,)\,\Gamma\,\|A\|\,\|\bar{Z}_k\|\Big)\Big] + \mathcal{O}(\varepsilon^{\,2}).\end{aligned} \tag{3.32}$$

Therefore, if Γ is not too big and $\|\bar{x}_0\| \approx \|\bar{x}_k\|$ then we have normwise backward stability.

REMARK 3.5. *We point out that a similar analysis can be made for GMRES, obtaining bounds better than those in [1]. However, if we use double precision ($f(\varepsilon\,) = \varepsilon\,$) during the forward and backward substitution in the solution of (3.20) and if $\Gamma \approx 1$, the improved bound on the residual for GMRES shows that GMRES is as stable as FGMRES because in the bound (3.22) we have $f(\varepsilon\,) = \varepsilon\,$. Unfortunately, for sparse Gaussian factorization the condition $\Gamma \approx 1$ is seldom satisfied as the results of [1] show.*

REMARK 3.6. *We observe that formulae (3.26) and (3.27) indicate that the algorithm could significantly benefit from a restarting procedure. We note that both GMRES and FGMRES restarting at each iteration are numerically equivalent to iterative refinement. Moreover, after each restart the norm of the new $\bar{x}_0$ is closer to the final $\|\bar{x}_k\|$.*

REMARK 3.7. *Of course, we should point out that (3.29) and (3.31) provide sufficient conditions for backward stability. That they are not necessary is seen from our numerical results on a very ill-conditioned problem (bcsstk20 in Table 6.1) where our approach is still very successful. We point out that for all our sparse cases the norm of $\bar{Z}_k$ is not too big because the value of k is quite small.*

4. Construction of test matrices. We generate test matrices with specified condition numbers and singular value distributions by a standard technique. First, we generate a diagonal matrix with the required properties, and then we pre- and post-multiply it by random orthogonal matrices. Thus, if we choose the matrix D to be $\mathrm{diag}\{d_i\}$, where

$$d_i = 10^{-c\left(\frac{i-1}{n-1}\right)^{\gamma}}, \tag{4.1}$$

ETNA
Kent State University
etna@mcs.kent.edu

then the singular values lie between 1 and 10^{-c}, the condition number is 10^{c}, and the singular value distribution is skewed by altering γ: $\gamma = 1$ gives a log-linear uniform distribution, $\gamma > 1$ gives a distribution skewed toward one, and $\gamma < 1$ gives a distribution skewed toward 10^{-c}. We then use MATLAB in a standard fashion to generate random orthogonal matrices H and V and run our factorization and solution algorithms on the matrix

$$A = HDV. \tag{4.2}$$

Prob. #	Total/Inner it	RR	$\|A\bar{Z}_{\hat{k}}\|$	$\| \|\bar{Z}_{\hat{k}}\| \|\bar{y}_{\hat{k}}\| \|$
1	14/14	9.6e-17	1.5e+00	2.0e+01
2	13/13	1.0e-16	1.5e+00	1.8e+01
3	14/14	4.8e-17	1.7e+00	2.0e+01
4	14/14	1.3e-16	1.5e+00	2.2e+01
5	14/14	9.0e-17	1.7e+00	1.9e+01
6	14/14	9.7e-17	1.6e+00	2.2e+01
7	14/14	6.7e-17	1.6e+00	2.0e+01
8	13/13	7.3e-17	1.5e+00	1.9e+01
9	13/13	5.7e-17	1.4e+00	1.8e+01
10	13/13	1.1e-16	1.4e+00	1.9e+01

TABLE 5.1
Random dense matrices. $n = 200$, $c = 8.2$, $\gamma = 0.5$, *and backward and forward substitutions in single precision.*

Prob. #	Total/Inner it	RR	$\|A\bar{Z}_{\hat{k}}\|$	$\| \|\bar{Z}_{\hat{k}}\| \|\bar{y}_{\hat{k}}\| \|$
1	14/14	8.3e-17	1.5e+00	2.0e+01
2	13/13	1.0e-16	1.5e+00	1.8e+01
3	12/12	1.5e-16	1.6e+00	1.9e+01
4	13/13	1.2e-16	1.5e+00	2.1e+01
5	14/14	7.2e-17	1.5e+00	1.9e+01
6	14/14	1.1e-16	1.5e+00	2.0e+01
7	14/14	4.8e-17	1.6e+00	2.0e+01
8	13/13	8.8e-17	1.5e+00	1.9e+01
9	13/13	6.0e-17	1.4e+00	1.8e+01
10	13/13	9.1e-17	1.3e+00	1.8e+01

TABLE 5.2
Random dense matrices. $n = 200$, $c = 8.2$, $\gamma = 0.5$, *and backward and forward substitutions in double precision.*

5. Experimental results. In this section we report on our experiments on dense unsymmetric matrices generated as described in Section 4. We conduct these experiments using MATLAB. We perform the single-precision factorization using `SGETRF` from LAPACK. More precisely, given the randomly generated matrix A as in equation (4.2), we use the MATLAB command

```
[L,U] = lu(single(A))
```

in order to generate the single-precision factors L and U. As mentioned in Section 1, we will use this single-precision factorization as a preconditioner for Richardson's method (that is iterative refinement) or FGMRES.

We present two variants of the preconditioning:

1. the vector $\bar{z}_k$ is computed using the forward and backward substitution algorithm in single precision on the single-precision conversion of vector $\bar{v}_k$,
2. the vector $\bar{z}_k$ is computed using the forward and backward substitution algorithm in double precision on $\bar{v}_k$ after we converted the factors L and U to double precision.

Prob. #	Total/Inner it	RR	$\|A\bar{Z}_{\hat{k}}\|$	$\| \|\bar{Z}_{\hat{k}}\| \|\bar{y}_{\hat{k}}\| \|$
1	26/26	2.5e-16	7.4e+00	1.9e+02
2	27/27	6.6e-16	4.2e+00	4.7e+02
3	25/25	1.7e-16	3.3e+00	5.9e+01
4	52/52	3.9e-15	4.6e+01	3.0e+03
	88/36	1.1e-16	4.6e+01	6.0e-04
5	24/24	1.3e-16	2.0e+00	3.8e+01
6	31/31	2.5e-16	8.8e+00	1.7e+02
7	24/24	2.0e-16	3.5e+00	1.2e+02
8	24/24	1.8e-16	2.7e+00	8.8e+01
9	26/26	2.7e-16	3.2e+00	1.5e+02
10	44/44	5.7e-16	1.9e+01	5.9e+02

TABLE 5.3
Random dense matrices. $n = 200$, $c = 8.2$, $\gamma = 1$, *and backward and forward substitutions in single precision.*

Prob. #	Total/Inner it	RR	$\|A\bar{Z}_{\hat{k}}\|$	$\| \|\bar{Z}_{\hat{k}}\| \|\bar{y}_{\hat{k}}\| \|$
1	20/20	1.7e-16	8.0e+00	7.3e+01
2	20/20	2.0e-16	3.9e+00	5.9e+01
3	20/20	2.7e-16	3.5e+00	4.0e+01
4	20/20	1.1e-15	4.5e+01	4.7e+02
	25/5	1.5e-16	4.8e+01	1.5e-05
5	20/20	2.6e-16	2.2e+00	2.8e+01
6	20/20	1.9e-16	1.1e+01	8.4e+01
7	20/20	2.0e-16	3.9e+00	6.9e+01
8	20/20	6.2e-16	3.0e+00	5.8e+01
9	20/20	3.2e-16	3.5e+00	3.6e+01
10	20/20	4.0e-16	2.0e+01	1.6e+02

TABLE 5.4
Random dense matrices. $n = 200$, $c = 8.2$, $\gamma = 1$, *and backward and forward substitutions in double precision.*

Note that all other computations are in double precision. The second case has the disadvantage of using more memory but makes the algorithm more robust. Moreover, even if the number of restarts increases, the total number of iterations decreases significantly in some examples. Note that our problems are essentially singular in single precision (we take c in equation (4.1) to be 8.2), so we should not be surprised if sometimes many iterations are required for full convergence to a scaled residual (backward error) at double-precision accuracy.

In all the tables, the second column reports the total number of iterations and the number of iterations after the last restart (Total/Inner). These numbers are of course the same if no restarting is required. Our restart algorithm is automatic. We stop when the Arnoldi residual is at machine precision and restart only if the actual residual is not. Finally, in all the tables we denote by RR the value

$$RR = \frac{\|b - A\bar{x}_{\hat{k}}\|}{(\|A\| \, \|\bar{x}_{\hat{k}}\| + \|b\|)}.$$

In Tables 5.1–5.6, we show the numerical results for A of dimension 200. We point out that for values of γ in (4.1) less than 1, both variants of FGMRES (see Tables 5.1 and 5.2) converge rapidly even for a condition number greater than 10^8. Note that the bounding quantities of equation (3.7) that are shown in columns 4 and 5 are very reasonable. For the case $\gamma = 0.5$, the iterative refinement algorithm also usually converges, though in somewhat more

ETNA
Kent State University
etna@mcs.kent.edu

Prob. #	Total/Inner it	RR	$\|A\bar{Z}_{\hat{k}}\|$	$\| \, \|\bar{Z}_{\hat{k}}\| \, \|\bar{y}_{\hat{k}}\| \, \|$
1	200/200	3.6e-09	2.2e+01	3.5e+09
	247/47	2.0e-16	2.3e+01	3.4e+01
2	200/200	2.0e-09	4.9e+01	2.1e+09
	256/56	2.0e-16	4.9e+01	8.5e+00
3	131/131	8.7e-13	8.8e+02	9.4e+05
	253/122	2.0e-16	9.1e+02	9.4e-01
4	58/58	7.8e-15	1.1e+01	3.6e+03
	89/31	1.8e-16	1.2e+01	3.2e-05
5	108/108	4.2e-14	1.2e+02	3.8e+04
	195/87	1.7e-16	1.1e+02	9.2e-02
6	200/200	3.6e-09	2.3e+02	4.3e+09
	299/99	2.0e-16	2.3e+02	7.9e+01
7	200/200	4.9e-10	3.3e+03	6.9e+08
	338/138	7.2e-16	3.2e+03	6.5e+02
8	78/78	1.4e-14	2.8e+01	1.0e+04
	128/50	2.0e-16	3.0e+01	6.5e-04
9	79/79	2.7e-15	1.9e+01	2.3e+03
	117/38	2.0e-16	2.0e+01	9.3e-05
10	48/48	1.4e-15	7.6e+00	6.9e+02
	75/27	2.0e-16	7.9e+00	5.3e-06

TABLE 5.5

Random dense matrices. $n = 200$, $c = 8.2$, $\gamma = 2$, *and backward and forward substitutions in single precision.*

Prob. #	Total/Inner it	RR	$\|A\bar{Z}_{\hat{k}}\|$	$\| \, \|\bar{Z}_{\hat{k}}\| \, \|\bar{y}_{\hat{k}}\| \, \|$
1	20/20	9.1e-11	6.1e+00	3.1e+02
	40/20	1.8e-15	6.4e+00	1.5e-01
	56/16	2.1e-16	6.1e+00	5.7e-06
2	20/20	2.5e-11	1.4e+01	7.9e+02
	40/20	9.5e-16	1.4e+01	9.5e-02
3	20/20	5.7e-11	8.2e+02	1.0e+05
	40/20	2.3e-16	7.2e+02	2.1e+01
4	20/20	4.7e-12	1.4e+01	1.9e+02
	39/19	2.0e-16	1.2e+01	1.8e-02
5	20/20	2.0e-11	1.2e+02	1.3e+03
	40/20	2.2e-16	1.2e+02	3.9e-01
6	20/20	4.2e-10	7.0e+01	1.4e+04
	40/20	8.0e-15	7.2e+01	1.6e+01
	56/16	2.2e-16	6.7e+01	1.3e-04
7	20/20	1.1e-12	8.8e+02	1.3e+04
	38/18	1.9e-16	9.1e+02	1.7e-01
8	20/20	6.1e-12	3.1e+01	1.1e+03
	40/20	2.0e-16	2.9e+01	5.3e-02
9	20/20	4.4e-12	1.9e+01	5.5e+02
	39/19	2.0e-16	2.1e+01	5.4e-02
10	20/20	5.5e-13	9.6e+00	1.7e+02
	36/16	2.1e-16	1.0e+01	2.6e-03

TABLE 5.6

Random dense matrices. $n = 200$, $c = 8.2$, $\gamma = 2$, *and backward and forward substitutions in double precision.*

iterations. For $\gamma = 1$, however, iterative refinement either does not converge or converges very slowly. We see in Tables 5.3 and 5.4 that both FGMRES variants converge and, although the bounding quantities are larger than in Tables 5.1 and 5.2, they are still reasonable, except for the fourth problem. Here we need to restart to get convergence to full precision. In the last case, when $\gamma = 2$, the behavior of all our algorithms deteriorates, and both FGMRES variants restart after we detect a small residual for the least-squares internal problem but the computed residual $\|b - A\bar{x}_k\| > \varepsilon$ ($\bar{x}_k$ is the computed solution). However, both variants converge after a few restarts (see Tables 5.5 and 5.6). Again, on convergence, the bounding quantities are reasonable.

We also tested our algorithm for A of dimension 400. The increased dimensionality of the matrix and the log-linear uniform distribution of the eigenvalues that causes a greater clustering near 10^{-c} exacerbates some of the behavior observed for the lower dimensional case. Although the $\gamma = 0.5$ distribution still works well (also for iterative refinement), the algorithms require more iterations (and restarts) as γ increases, although we eventually converge to double-precision machine precision.

We also computed the value of $\|\bar{Z}_k\|$, as this appeared in our bounds derived in [1]. We found that, although $\|\bar{Z}_k\|$ had a similar value to $\|\,|\bar{Z}_k|\,|\bar{y}_k|\,\|$ for the first few iterations, it rapidly became much larger and was typically 10^7 times larger than $\|\,|\bar{Z}_k|\,|\bar{y}_k|\,\|$ for large values of k ($k > 30$).

6. Extension to sparse systems. We cannot extend the experimental results to sparse systems totally within MATLAB, since our version of MATLAB computes a sparse factorization only in double precision. We thus compute the factors separately in a Fortran program using a single-precision version of `MA57` and then convert these to data structures that we can feed directly to MATLAB that performs the rest of the computation in double precision. We present the results for a selection of sparse problems in Table 6.1. For all these test matrices, $\Gamma \gg 1$ because our factorization is in single precision and the condition numbers of the matrices are greater than $\varepsilon^{-1/2}$. Thus, the preconditioned matrix has a spectral radius close to 1 or greater than 1. This explains why iterative refinement converges very slowly if at all.

Although our main intention in this paper is to investigate the numerical feasibility of using mixed arithmetic, we show in Figure 6.1 a summary of results obtained by our colleagues Jonathan Hogg and Jennifer Scott who have developed an HSL code `HSL_MA79` that is based on the theory established in this paper. In the figure, we see for a subset of the Test 1 set described in [10] that, for problems that are reasonably large (in terms of double-precision factorization time), the mixed arithmetic approach is typically 1.5 times faster than using double precision and, of course, requires less storage. We refer the reader to [10] for further numerical results.

7. Conclusions. We have established both by theory and by experiments that solutions with a backward error at the double-precision level can be obtained when using a single-precision factorization that is used as a preconditioner for FGMRES. We have also found that iterative refinement often does not work in such cases. This implies that we can take advantage of the faster speed of single-precision arithmetic on machines where speed or storage considerations give advantages for this mode of working. We have illustrated that this applies to sparse matrix factorizations as well as in the dense case.

Furthermore, all our analysis would be equally valid if extended-precision accuracy was required from a double-precision factorization. In this case, the penalties of using extended precision, normally implemented in software, are very significant.

Finally, in [11, 12] an error analysis of other Krylov based algorithms without preconditioning is presented in detail. However, the theoretical results suggest that the ORTHOMIN

	Iterative refinement		FGMRES			
Matrix	Total it	RR	Total / inner	RR	$\Vert A\bar{Z}_{\hat{k}}\Vert$	$\Vert \, \vert\bar{Z}_{\hat{k}}\vert \, \vert\bar{y}_{\hat{k}}\vert \, \Vert$
bcsstk20 $n = 485$ $\kappa(A) \approx 4 \times 10^{12}$	30	2.1e-15	2 / 2	1.4e-11	1.7e+00	4.6e+02
			4 / 2	3.4e-14	1.6e+00	3.8e-01
			6 / 2	7.2e-17	1.6e+00	5.6e-04
bcsstm27 $n = 1224$ $\kappa(A) \approx 5 \times 10^{9}$	22	1.6e-15	2 / 2	5.8e-11	1.7e+00	2.7e+01
			4 / 2	1.8e-11	6.3e-01	1.3e+00
			6 / 2	6.0e-13	2.0e+00	7.6e-02
			8 / 2	1.5e-13	1.7e+00	1.0e-02
			10 / 2	1.2e-14	1.7e+00	1.9e-03
			12 / 2	2.6e-15	1.8e+00	1.7e-04
			14 / 2	1.8e-16	1.6e+00	4.3e-05
s3rmq4m1 $n = 5489$ $\kappa(A) \approx 4 \times 10^{9}$	16	2.2e-15	2 / 2	3.5e-11	1.0e+00	8.6e+01
			4 / 2	2.1e-13	1.1e+00	3.2e-01
			6 / 2	4.5e-15	1.7e+00	6.4e-03
			8 / 2	1.1e-16	1.6e+00	1.3e-04
s3dkq4m2 $n = 90449$ $\kappa(A) \approx 7 \times 10^{10}$	53	1.1e-10	10 / 10	6.3e-17	1.2e+00	1.2e+03

TABLE 6.1
Results for sparse matrices.

and GCR families of algorithms can be only conditionally stable even if in practice their behavior is quite satisfactory. Future work will involve numerical comparison between these methods and FGMRES.

Acknowledgments. We would like to thank the editor, Yousef Saad, and the anonymous referees for their valuable suggestions.

REFERENCES

[1] M. ARIOLI, I. S. DUFF, S. GRATTON, AND S. PRALET, *A note on GMRES preconditioned by a perturbed LDL^T decomposition with static pivoting*, SIAM J. Sci. Comput., 29 (2007), pp. 2024–2044.

[2] M. ARIOLI, *Roundoff error analysis of orthogonal factorizations of upper Hessenberg rectangular matrices*, Technical Report RAL-TR-2008-004, Rutherford Appledon Laboratory, Oxfordshire, 2008.

[3] Å. BJÖRCK, AND C. C. PAIGE, *Loss and recapture of orthogonality in the modified Gram–Schmidt algorithm*, SIAM J. Matrix Anal. Appl., 13 (1992), pp. 176–190.

[4] 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*, Int. J. of High Performance Computing Applications, 21 (2007), pp. 457–466.

[5] A. BUTTARI, J. DONGARRA, J. KURZAK, P. LUSZCZEK, AND S. TOMOV, *Using mixed precision for sparse matrix computations to enhance the performance while achieving 64-bit accuracy*, ACM Trans. Math. Software, 34 (2008), pp. 17:1–17:22.

[6] I. S. DUFF, *MA57 – A code for the solution of sparse symmetric indefinite systems*, ACM Trans. Math. Software 30 (2004), pp. 118–144.

[7] K. O. GEDDES AND W. W. ZHENG, *Exploiting fast hardware floating point in high precision computation* in Proceedings of the 2003 International Symposium on Symbolic and Algebraic Computation. Philadelphia, PA, USA, 2003, pp. 111–118.

[8] L. GIRAUD AND J. LANGOU, *When modified Gram-Schmidt generates a well-conditioned set of vectors*, IMA J. Numer. Anal., 22 (2002), pp. 521–528.

[9] N. J. HIGHAM, *Accuracy and Stability of Numerical Algorithms*, 2nd edition, Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2002.

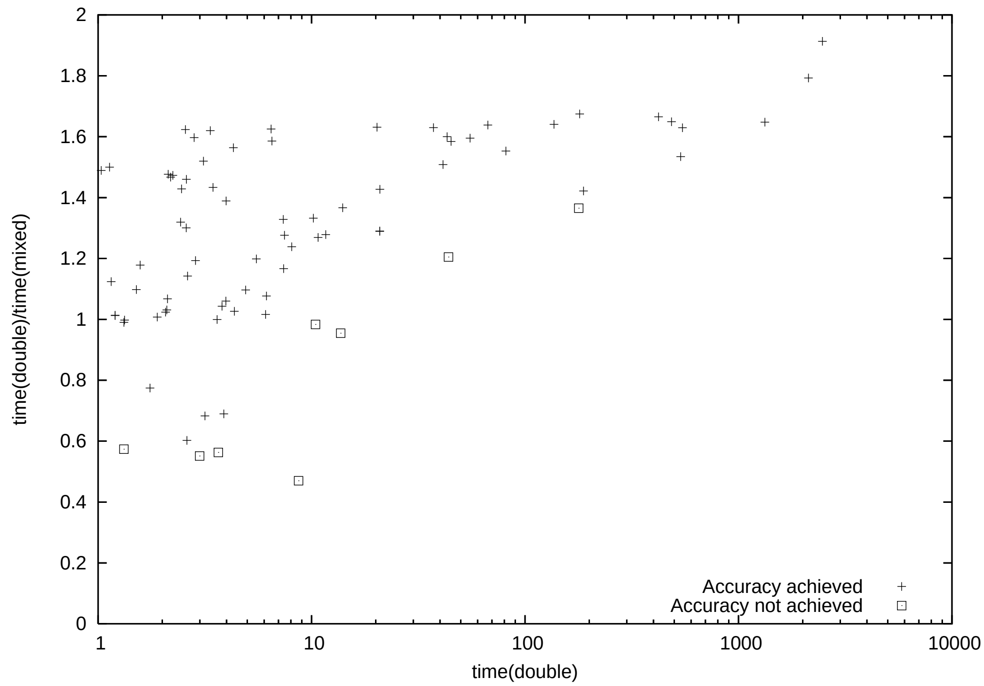

FIGURE 6.1. *Ratio of times to solve (1.1) in mixed precision and double precision on a test set of* 78 *sparse problems with a scaled residual* $\dfrac{\|b - A\bar{x}_{\hat{k}}\|}{(\|A\|\,\|\bar{x}_{\hat{k}}\| + \|b\|)} \leq 5 \times 10^{-15}$.

[10] J. D. HOGG AND J. A. SCOTT, *On the use of mixed precision for the fast and robust solution of sparse symmetric linear systems*, Technical Report RAL-TR-2008-023, Rutherford Appledon Laboratory, Oxfordshire, 2008.

[11] P. JIRÁNEK, *Limiting Accuracy of Iterative Methods*, PhD Thesis, Faculty of Mechatronics and Interdisiplinary Engineering Studies, Technical University of Liberec, and Institute of Computer Sciences, Academy of Sciences of the Czech Republic, 2007.

[12] P. JIRÁNEK, M. ROZLOŽNÍK, AND M. H. GUTKNECHT, *How to make simpler GMRES and GCR more stable*, SIAM J. Matrix Anal. Appl., 30 (2008), pp. 1483-1499.

[13] W. KAHAN, *Lecture Notes on the status of IEEE Standards 754 for Binary Floatin-Point Arithmetic*, http://http.cs.berkeley.edu/~wkahan/ieee754status/ieee754.ps.

[14] 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.

[15] Y. SAAD, *A flexible inner-outer preconditioned GMRES algorithm*, SIAM J. Sci. Stat. Comp., 14 (1993), pp. 461–469.

[16] Y. SAAD AND M. H. SCHULTZ, *GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems*. SIAM J. Sci. Stat. Comp., 7 (1986), pp. 856–869.

[17] R. D. SKEEL, *Iterative refinement implies numerical stability for Gaussian elimination*, Math. Comp., 35 (1980), pp. 817–832.

[18] J. H. WILKINSON, *The Algebraic Eigenvalue Problem*, Oxford University Press, Oxford, 1965.