VERIFIED SOLUTIONS OF LINEAR SYSTEMS WITHOUT DIRECTED ROUNDING

Technical Report No.2005-04

TAKESHI OGITA *SIEGFRIED M. RUMP* *SHIN'ICHI OISHI*

July,2005

Advanced Research Institute
for Science and Engineering,
Waseda University

VERIFIED SOLUTIONS OF LINEAR SYSTEMS WITHOUT DIRECTED ROUNDING *

TAKESHI OGITA †, SIEGFRIED M. RUMP ‡, AND SHIN'ICHI OISHI §

Abstract. This paper is concerned with the accuracy of an approximate solution to the exact solution of linear systems including the verification of nonsingularity of the system matrix. There are a number of self-validating algorithms to compute guaranteed error bounds on approximate solutions with directed rounding using the rounding mode controlled instructions defined by the IEEE 754 standard for floating-point arithmetic. In this paper we will propose a verification method without using directed rounding. That means, guaranteed error bounds on approximate solutions can be calculated only in round-to-nearest mode. Numerical results will be presented showing properties and limits of the proposed verification method.

Key words. self-validating methods, verified solution of linear systems, directed rounding, round-to-nearest

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

1. Introduction. We are concerned with the problem of verifying the accuracy of an approximate solution $\widetilde{x}$ of a linear system

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

where A is a real $n \times n$ matrix and b is a real n-vector. If A is nonsingular, there exists a unique exact solution $x^* := A^{-1}b$ and we are aiming on calculating some $\kappa \in \mathbb{R}$ with

$$\|\widetilde{x} - x^*\|_\infty \leq \kappa. \tag{1.2}$$

We also aim on verifying nonsingularity of A by the method. There are a number of self-validating algorithms (cf., for example, [9, 11]) to verify the nonsingularity of A and to compute some κ satisfying (1.2). These algorithms use directed roundings defined in IEEE 754 standard for floating-point arithmetic [1].

Most of today's digital computers support the IEEE 754. Using the rounding mode controlled instructions defined in the IEEE 754, self-validating algorithms have been implemented in numerical libraries, e.g. ACRITH-XSC [13], C-XSC [6], PROFIL/BIAS [7] and INTLAB [10]. Among them INTLAB, which is developed by the second author, is a fast Matlab toolbox. Using INTLAB, we can easily switch the rounding mode by the `setround` function. The `setround` function has already been a built-in function as

```
system_dependent('setround',mode)
```

on Matlab version 5.3 or later [12] under Windows OS. A good introduction into INTLAB is [2].

However, we sometimes encounter situations where the rounding mode in floating-point arithmetic cannot be switched in the verification process because of the compiler or programming language in use. For example, most of FORTRAN77 compilers (e.g. g77 in GNU Compiler Collection) do not support it. Moreover, the lack of floating-point arithmetic in Java has been pointed out by Kahan and Darcy [4]. In such situations, we are forced to use partially the object files created by a different compiler which can realize the switch of rounding mode. For example, one may use the Java Native Interface (JNI) to handle those situations. Furthermore, the implementation of directed rounding is still not standardized even if the compiler in use

*This research was partially supported by Core Research for Evolutional Science and Technology (CREST) program, Japan Science and Technology Agency (JST).

†CREST, JST and Faculty of Science and Engineering, Waseda University, Tokyo 169-0072, Japan (ogita@waseda.jp).

‡Institut für Informatik III, Technische Universität Hamburg-Harburg, Schwarzenbergstraße 95, Hamburg 21071, Germany (rump@tu-harburg.de).

§Faculty of Science and Engineering, Waseda University and CREST, JST, Tokyo 169-0072, Japan (oishi@waseda.jp).

supports it. We often use the different compilers on several platform, so that the way to call the function of controlling the rounding mode is also different. Therefore, programs using directed rounding become less portable. To maintain the portability with FORTRAN77, the INTLIB [5], which is developed by Kearfott et al., simulates the directed rounding by multiplying computed results by $(1 \pm \epsilon)$ for an appropriate $\epsilon \geq 0$. However, such simulations of the directed rounding sometimes slow down computations significantly.

A main point of this paper is to show that we can verify the nonsingularity of A and to compute an error κ satisfying (1.2) *without* directed rounding. We suppose that all floating-point operations are executed in round-to-nearest mode. Under such condition, we will propose a fast self-validating algorithm for linear systems. Moreover, it is intended to achieve the computational speed using a priori error estimates. Numerical results will be presented showing properties and limits of the proposed verification method.

2. Floating-point arithmetic. We start by stating some well-known properties on floating-point numbers. Let $\mathbb{R}$ denote the set of real numbers. Let $\mathbb{F}$ be a set of floating-point numbers abiding by IEEE 754 standard. It is well known that $\mathbb{F}$ is symmetric, i.e., $x \in \mathbb{F} \Rightarrow -x \in \mathbb{F}$, so that $|x|$ is exact for $x \in \mathbb{F}$. Throughout this paper, we assume that the operations in $\mathrm{fl}(\cdot)$ is all executed by floating-point arithmetic in given rounding mode (default is round-to-nearest). Throughout the paper we assume that no overflow occurs. This usually leads to a premature stop of calculations, so we do not have to check for this. Furthermore, we assume that the system in this paper supports the gradual underflow, which is a requirement of IEEE 754.

Let $\mathbf{u}$ and $\underline{\mathbf{u}}$ be the unit roundoff and the underflow unit, respectively (especially, $\mathbf{u} = 2^{-53}$ and $\underline{\mathbf{u}} = 2^{-1074}$ in IEEE 754 double precision). Here, we denote the smallest (positive) normalized floating-point number by $\mathbf{u}_{\mathrm{N}} := \underline{\mathbf{u}} \cdot \mathbf{u}^{-1}$. For $a, b \in \mathbb{F}$

$$a + b = (1 + \varepsilon)\mathrm{fl}(a + b) \quad \text{for } |\varepsilon| \leq \mathbf{u}, \quad \text{and} \tag{2.1}$$

$$a \cdot b = (1 + \delta)\mathrm{fl}(a \cdot b) + \eta \quad \text{for } |\delta| \leq \mathbf{u} \text{ and } |\eta| \leq \underline{\mathbf{u}}. \tag{2.2}$$

For $a, b \in \mathbb{F}$, it holds that

$$(1 - \mathbf{u})\,|\mathrm{fl}(a \circ b)| \leq |a \circ b| \leq (1 + \mathbf{u})\,|\mathrm{fl}(a \circ b)| \quad \text{for } \circ \in \{+, -\} \tag{2.3}$$

$$(1 - \mathbf{u})\,|\mathrm{fl}(a \circ b)| - \underline{\mathbf{u}} \leq |a \circ b| \leq (1 + \mathbf{u})\,|\mathrm{fl}(a \circ b)| + \underline{\mathbf{u}} \quad \text{for } \circ \in \{\cdot, /\} \tag{2.4}$$

including possible underflow. Obviously, it also holds for $\circ \in \{\cdot, /\}$ that

$$|a| \circ |b| \leq (1 + \mathbf{u}) \max(\mathrm{fl}(|a| \circ |b|), \mathbf{u}_{\mathrm{N}}). \tag{2.5}$$

For later use, we cite here some definitions.

Definition 2.1. *The constants γ_n and $\widetilde{\gamma}_n$ for $n \in \mathbb{N}$ are defined by*

$$\gamma_n := \frac{n\mathbf{u}}{1 - n\mathbf{u}} \quad \textit{and} \quad \widetilde{\gamma}_n := \mathrm{fl}\left(\frac{n\mathbf{u}}{1 - n\mathbf{u}}\right). \tag{2.6}$$

When using γ_n and $\widetilde{\gamma}_n$, we implicitly assume that $n\mathbf{u} < 1$.

Definition 2.2. *The constants ζ_n for $n \in \mathbb{N}$ is defined by*

$$\zeta_n := (1 + \mathbf{u})^n - 1. \tag{2.7}$$

We present some fundamental properties relating $\mathbf{u}$, $\underline{\mathbf{u}}$, γ_n, $\widetilde{\gamma}_n$ and ζ_n. Some of them have already been presented in Higham's excellent book [3, e.g. Lemma 3.3].

LEMMA 2.3. *Let* $\mathbf{u}$ *and* $\underline{\mathbf{u}}$ *be the unit roundoff and the underflow unit. Let* $\widetilde{\gamma}_n$ *and* ζ_n *for* $n \in \mathbb{N}$ *be as in Definition 2.1 and Definition 2.2. The following relations hold:*

$$(1+\mathbf{u})^n \le \frac{1}{(1-\mathbf{u})^n} \le \frac{1}{1-n\mathbf{u}} \tag{2.8}$$

$$(1+\mathbf{u})^p \zeta_n \le \zeta_{n+p} \tag{2.9}$$

$$\zeta_n + \zeta_p \le \zeta_{n+p} \tag{2.10}$$

$$\zeta_n \le \widetilde{\gamma}_n \tag{2.11}$$

Moreover, the following relations hold for $a \in \mathbb{F}$*:*

$$\frac{|a|}{1-n\mathbf{u}} \le \mathrm{fl}\left(\frac{\max(|a|, \mathbf{u}_{\mathrm{N}})}{1-(n+1)\mathbf{u}}\right) \tag{2.12}$$

$$\zeta_n |a| \le \mathrm{fl}\left(\widetilde{\gamma}_{n+1} \max(|a|, \mathbf{u}_{\mathrm{N}})\right) \tag{2.13}$$

where $\mathbf{u}_{\mathrm{N}} := \underline{\mathbf{u}} \cdot \mathbf{u}^{-1}$*. In particular, for* $\mathbf{u}_{\mathrm{N}} \le |a| \in \mathbb{F}$

$$\frac{|a|}{1-n\mathbf{u}} \le \mathrm{fl}\left(\frac{|a|}{1-(n+1)\mathbf{u}}\right) \tag{2.14}$$

$$\zeta_n |a| \le \mathrm{fl}\left(\widetilde{\gamma}_{n+1} |a|\right). \tag{2.15}$$

Proof. The proof of Lemma 2.3 is deferred to Appendix A. □

Some of the usual formulas for γ_n apply to ζ_n like (2.9) and (2.10). We can check more. However, it seems not true that

$$\frac{\zeta_n}{(1-\mathbf{u})^m} \le \zeta_{m+n}.$$

For a real matrix $A = [a_{ij}]$, we denote by $|A|$ the nonnegative matrix consisting of entrywise absolute values $|a_{ij}|$. For a real vector x, we apply this definition similarly. We denote the maximum norm of a real n-vector $x = (x_1, \ldots, x_n)^T$ and a real $m \times n$ matrix $A = [a_{ij}]$ by

$$\|x\|_\infty := \max_{1\le i\le n} |x_i| \quad \text{and} \quad \|A\|_\infty := \max_{1\le i\le m} \sum_{j=1}^{n} |a_{ij}|,$$

respectively.

The following is a well-known result for floating-point computation of a summation.

LEMMA 2.4 (e.g. Higham [3]). *For* $p \in \mathbb{F}^n$*, the following inequality holds including the presence of underflow:*

$$\left| \mathrm{fl}\left(\sum_{i=1}^{n} p_i\right) - \sum_{i=1}^{n} p_i \right| \le \gamma_{n-1} \sum_{i=1}^{n} |p_i|. \tag{2.16}$$

Lemma 2.4 can slightly be improved into Lemma 2.5 by introducing a new concept $\zeta_n := (1+\mathbf{u})^n - 1$ instead of γ_n.

LEMMA 2.5. *For* $p \in \mathbb{F}^n$*, the following inequality holds including the presence of underflow:*

$$\left| \mathrm{fl}\left(\sum_{i=1}^{n} p_i\right) - \sum_{i=1}^{n} p_i \right| \le \zeta_{n-1} \sum_{i=1}^{n} |p_i|. \tag{2.17}$$

We also present the following lemma for dot product.

LEMMA 2.6. *For $x, y \in \mathbb{F}^n$, the following inequality holds including the presence of underflow:*

$$\left|\mathrm{fl}\left(x^T y\right) - x^T y\right| \leq \zeta_n |x^T|\,|y| + n\underline{\mathbf{u}}. \tag{2.18}$$

Note that (2.17) and (2.18) are independent of the order of execution of floating-point additions and multiplications. From this, we have the following theorem.

THEOREM 2.7. *For $x, y \in \mathbb{F}^n$, the following inequality holds including the presence of underflow:*

$$\left|\mathrm{fl}\left(x^T y\right) - x^T y\right| \leq \mathrm{fl}\left(\widetilde{\gamma}_{2n+1} |x^T|\,|y|\right) + (n+1)\underline{\mathbf{u}}. \tag{2.19}$$

For later use, we present some relations between exact and floating-point vector and matrix operations.

LEMMA 2.8. *Let $x, y \in \mathbb{F}^n$ and $A \in \mathbb{F}^{m\times n}$. Let $e = (1, \ldots, 1)^T \in \mathbb{R}^n$. Then the following relations hold:*

$$\sum_{i=1}^{n} |x_i| \leq (1+\mathbf{u})^{n-1} \mathrm{fl}\left(\sum_{i=1}^{n} |x_i|\right) \tag{2.20}$$

$$\left|x^T\right|\,|y| \leq (1+\mathbf{u})^n \mathrm{fl}\left(\left|x^T\right|\,|y|\right) + n\underline{\mathbf{u}} \tag{2.21}$$

$$\|x\|_\infty = \mathrm{fl}\left(\|x\|_\infty\right) \tag{2.22}$$

$$|A|\,e \leq (1+\mathbf{u})^{n-1} \mathrm{fl}\left(|A|\,e\right) \tag{2.23}$$

$$|A|\,|x| \leq (1+\mathbf{u})^n \mathrm{fl}\left(|A|\,|x|\right) + n\underline{\mathbf{u}} \cdot e \tag{2.24}$$

$$\|A\|_\infty \leq (1+\mathbf{u})^{n-1} \mathrm{fl}\left(\|A\|_\infty\right). \tag{2.25}$$

Proof. We prove only (2.25), the other statements follow similarly. By definition of the maximum norm we have

$$\|A\|_\infty = \||A|\,e\|_\infty.$$

By (2.22) and (2.23), it follows that

$$\||A|\,e\|_\infty \leq \left\|(1+\mathbf{u})^{n-1} \mathrm{fl}\left(|A|\,e\right)\right\|_\infty = (1+\mathbf{u})^{n-1} \left\|\mathrm{fl}\left(|A|\,e\right)\right\|_\infty = (1+\mathbf{u})^{n-1} \mathrm{fl}\left(\||A|\,e\|_\infty\right),$$

which implies (2.25). □

3. Verification theory. We present in the following a fundamental theorem to calculate the error bounds on approximate solutions of systems of linear equations.

THEOREM 3.1 (e.g. Oishi and Rump [9]). *Let A be a real $n \times n$ matrix and b an real n-vector. Let $\widetilde{x}$ be an approximate solution of $Ax = b$ and R a preconditioner of A. If*

$$\|RA - I\|_\infty < 1 \tag{3.1}$$

for I denoting the $n \times n$ identity matrix, then A is nonsingular and

$$\left\|\widetilde{x} - A^{-1}b\right\|_\infty \leq \frac{\|R(A\widetilde{x} - b)\|_\infty}{1 - \|RA - I\|_\infty}. \tag{3.2}$$

A good choice of R may be an approximate inverse of A. We want to calculate an upper bound on $\|RA - I\|_\infty$ and on $\|R(A\widetilde{x} - b)\|_\infty$ such that $\alpha, \beta \in \mathbb{F}$ and

$$\|RA - I\|_\infty \le \alpha \quad \text{and} \quad \|R(A\widetilde{x} - b)\|_\infty \le \beta, \tag{3.3}$$

respectively. If $\alpha < 1$, then A is nonsingular and

$$\left\|\widetilde{x} - A^{-1}b\right\|_\infty \le \frac{\beta}{1-\alpha}. \tag{3.4}$$

After having α and β, we can calculate the error bound as follows:

THEOREM 3.2. *Let A, b and $\widetilde{x}$ be as in Theorem 3.1, and let α and β satisfy (3.3) and $\alpha, \beta \in \mathbb{F}$. If $\alpha < 1$, then A is nonsingular and*

$$\left\|\widetilde{x} - A^{-1}b\right\|_\infty \le \mathrm{fl}\left(\left(\frac{\max(\beta, \mathbf{u}_{\mathrm{N}})}{1-\alpha}\right)/(1-3\mathbf{u})\right) \tag{3.5}$$

including the possible underflow.

Proof. By (2.3), (2.5) and (2.8), it follows that

$$\frac{\beta}{1-\alpha} \le \frac{\beta}{(1-\mathbf{u})\cdot \mathrm{fl}\,(1-\alpha)} \le \frac{1+\mathbf{u}}{1-\mathbf{u}} \cdot \mathrm{fl}\left(\frac{\max(\beta, \mathbf{u}_{\mathrm{N}})}{1-\alpha}\right) \le \frac{1}{(1-\mathbf{u})^2} \cdot \mathrm{fl}\left(\frac{\max(\beta, \mathbf{u}_{\mathrm{N}})}{1-\alpha}\right).$$

From (2.8) and (2.12), we have the desired formula (3.5). ∎

From (3.5), we see that any value α less than $1/2$, say, is sufficient for a reasonable error estimation. So not too much effort must be spent on estimating α. On the other hand, β is directly contributing to the quality of the estimation, so that we should especially pay attention to the estimation for β.

4. Standard estimation. In this section, we present self-validating algorithms to calculate an upper bound on $\left\|\widetilde{x} - A^{-1}b\right\|_\infty$ using a standard estimation.

For comparison of the algorithms with or without directed rounding, we shall first present standard algorithms to calculate upper bounds on $\|RA - I\|_\infty$ and $\|R(A\widetilde{x} - b)\|_\infty$ *with* directed rounding. Throughout this paper, we express algorithms in Matlab-style, which is almost executable INTLAB code.

ALGORITHM 4.1 (Oishi and Rump [9]). *Standard calculation of α and β such that $\|RA - I\|_\infty \le \alpha$ and $\|R(A\widetilde{x} - b)\|_\infty \le \beta$ with directed rounding.*

function $[\alpha, \beta] = \mathrm{std.rnd}(A, b, \widetilde{x})$

Compute a preconditioner R by a favorite algorithm	% *for example,* $R = \mathtt{inv}(A)$
setround(-1)	% *round-downward mode*
$\underline{G} = \mathrm{fl}\,(RA - I)\,; \quad \underline{r} = \mathrm{fl}\,(A\widetilde{x} - b)\,;$	% *lower bound for $RA - I$ and $A\widetilde{x} - b$*
setround$(+1)$	% *round-upward mode*
$\overline{G} = \mathrm{fl}\,(RA - I)\,; \quad \overline{r} = \mathrm{fl}\,(A\widetilde{x} - b)\,;$	% *upper bound for $RA - I$ and $A\widetilde{x} - b$*
$\alpha = \mathrm{fl}\left(\left\|\max(\lvert\underline{G}\rvert, \lvert\overline{G}\rvert)\right\|_\infty\right)\,;$	% $\alpha \ge \|RA - I\|_\infty$
$r_{\mathrm{mid}} = (\underline{r} + \overline{r})/2; \quad r_{\mathrm{rad}} = r_{\mathrm{mid}} - \underline{r};$	% *midpoint and radius of $A\widetilde{x} - b$*
setround(-1)	
$\underline{t} = \mathrm{fl}\,(R \cdot r_{\mathrm{mid}})\,;$	% *lower bound for $R \cdot r_{\mathrm{mid}}$*
setround$(+1)$	
$\overline{t} = \mathrm{fl}\,(R \cdot r_{\mathrm{mid}})\,;$	% *upper bound for $R \cdot r_{\mathrm{mid}}$*
$\beta = \mathrm{fl}\left(\left\|\max(\lvert\underline{t}\rvert, \lvert\overline{t}\rvert) + \lvert R\rvert \cdot r_{\mathrm{rad}}\right\|_\infty\right)\,;$	% $\beta \ge \|R(A\widetilde{x} - b)\|_\infty$

Note that Algorithm 4.1 is still valid in the presence of underflow. The algorithm shows that calculation of α and β is considerably easy if directed rounding can be used. If an LU factorization (with partial pivoting) has been excuted, we can compute R by some algorithms (e.g. LINPACK's `DGEDI` and LAPACK's `DGETRI`) in $\frac{4}{3}n^3$ flops. Therefore, Algorithm 4.1 requires $\frac{16}{3}n^3$ flops in total. On the other hand, an LU factorization (or calculation of an approximate solution of $Ax = b$) requires $\frac{2}{3}n^3$ flops, so that the theoretical ratio of computational cost is 8.

We next propose standard algorithms *without* directed rounding. Using Lemma 2.6 and considering the presence of underflow, it holds for $e = (1, \dots, 1)^T$ that

$$|RA - I| \le \mathrm{fl}\left(|RA - I|\right) + \zeta_{n+1}(|R|\,|A| + I) + n\underline{\mathbf{u}} \cdot ee^T.$$

Therefore,

$$\|RA - I\|_\infty \le \|\mathrm{fl}\left(RA - I\right)\|_\infty + \zeta_{n+1}(\||R|\,|A|\|_\infty + 1) + n^2\underline{\mathbf{u}}. \tag{4.1}$$

By (2.25), we have

$$\|\mathrm{fl}\left(RA - I\right)\|_\infty \le (1+\mathbf{u})^{n-1}\alpha_1, \tag{4.2}$$

where $\alpha_1 := \mathrm{fl}\left(\|RA - I\|_\infty\right)$. By (2.24) and (2.23), it follows that

$$\begin{aligned} |R|\,|A|\,e &\le (1+\mathbf{u})^{n-1}\,|R|\,\mathrm{fl}\left(|A|\,e\right) \\ &\le (1+\mathbf{u})^{n-1}(1+\mathbf{u})^{n}\mathrm{fl}\left(|R|\,(|A|\,e)\right) + n(1+\mathbf{u})^{n-1}\underline{\mathbf{u}} \cdot e \\ &\le (1+\mathbf{u})^{2n-1}\mathrm{fl}\left(|R|\,(|A|\,e)\right) + n(1+\mathbf{u})^{n-1}\underline{\mathbf{u}} \cdot e. \end{aligned}$$

From this and (2.22), it holds that

$$\||R|\,|A|\|_\infty = \||R|\,|A|\,e\|_\infty \le (1+\mathbf{u})^{2n-1}\alpha_2 + n(1+\mathbf{u})^{n-1}\underline{\mathbf{u}}, \tag{4.3}$$

where $\alpha_2 := \mathrm{fl}\left(\||R|\,(|A|\,e)\|_\infty\right)$. Inserting (4.2) and (4.3) to (4.1), it follows that

$$\begin{aligned} \|RA - I\|_\infty &\le (1+\mathbf{u})^{n-1}\alpha_1 + \zeta_{n+1}\left\{(1+\mathbf{u})^{2n-1}\alpha_2 + n(1+\mathbf{u})^{n-1}\underline{\mathbf{u}} + 1\right\} + n^2\underline{\mathbf{u}} \\ &\le (1+\zeta_{n-1})\alpha_1 + \zeta_{3n}\alpha_2 + \zeta_{n+1} + n(n+\zeta_{2n})\underline{\mathbf{u}}. \end{aligned}$$

If $\alpha_1 < 1$, then $\zeta_{n-1}\alpha_1 < \zeta_{n-1}$. From this and using $n(n+\zeta_{2n})\underline{\mathbf{u}} \le \zeta_{2n}$, we have

$$\begin{aligned} \|RA - I\|_\infty &\le \alpha_1 + \zeta_{n-1} + \zeta_{3n}\alpha_2 + \zeta_{n+1} + \zeta_{2n} \\ &\le \alpha_1 + \zeta_{3n}\alpha_2 + 2\zeta_{2n} \le \alpha_1 + \zeta_{3n}(\alpha_2 + 2) \\ &\le \alpha_1 + (1+\mathbf{u})^{-1}\zeta_{3n+2}\mathrm{fl}\left(\alpha_2 + 2\right) \\ &\le \alpha_1 + \mathrm{fl}\left(\widetilde{\gamma}_{3n+2}(\alpha_2 + 2)\right). \end{aligned}$$

This proves the following theorem:

THEOREM 4.2. *Let A and R be a real $n \times n$ matrix and some preconditioner, respectively. Assume $(3n+2)\mathbf{u} < 1$. Let α_1 and α_2 be defined by*

$$\alpha_1 := \mathrm{fl}\left(\|RA - I\|_\infty\right) \quad \textit{and} \quad \alpha_2 := \mathrm{fl}\left(\||R|\,(|A|\,e)\|_\infty\right).$$

If $\alpha_1 < 1$, then the following inequality holds including the presence of underflow:

$$\|RA - I\|_\infty \le \mathrm{fl}\left(\frac{\alpha_1 + \mathrm{fl}\left(\widetilde{\gamma}_{3n+2}(\alpha_2 + 2)\right)}{1 - 2\mathbf{u}}\right). \tag{4.4}$$

We next evaluate $\|R(A\widetilde{x}-b)\|_\infty$. We assume that $r_{\mathrm{mid}}, r_{\mathrm{rad}} \in \mathbb{F}^n$ and $r_{\mathrm{rad}} \geq \mathbf{0}$ satisfying

$$r_{\mathrm{mid}} - r_{\mathrm{rad}} \leq A\widetilde{x} - b \leq r_{\mathrm{mid}} + r_{\mathrm{rad}} \tag{4.5}$$

are given. From (4.5), it follows that

$$|R(A\widetilde{x}-b)| \leq |R \cdot r_{\mathrm{mid}}| + |R| \cdot r_{\mathrm{rad}}. \tag{4.6}$$

Using (2.18),

$$|R \cdot r_{\mathrm{mid}}| \leq \mathrm{fl}\left(|R \cdot r_{\mathrm{mid}}|\right) + \zeta_n |R| \cdot |r_{\mathrm{mid}}| + n\underline{\mathbf{u}} \cdot e. \tag{4.7}$$

Inserting (4.7) into (4.6), we have

$$|R(A\widetilde{x}-b)| \leq \mathrm{fl}\left(|R \cdot r_{\mathrm{mid}}|\right) + |R|\left(\zeta_n |r_{\mathrm{mid}}| + r_{\mathrm{rad}}\right) + n\underline{\mathbf{u}} \cdot e. \tag{4.8}$$

Here,

$$\zeta_n |r_{\mathrm{mid}}| \leq \mathrm{fl}\left(\widetilde{\gamma}_{n+1} \max(|r_{\mathrm{mid}}|, \mathbf{u}_{\mathrm{N}} \cdot e)\right) =: t. \tag{4.9}$$

Therefore,

$$\begin{aligned}
|R|\left(\zeta_n |r_{\mathrm{mid}}| + r_{\mathrm{rad}}\right) + n\underline{\mathbf{u}} \cdot e &\leq |R|\left(t + r_{\mathrm{rad}}\right) \\
&\leq (1+\mathbf{u})\,|R| \cdot \mathrm{fl}\left(t + r_{\mathrm{rad}}\right) + n\underline{\mathbf{u}} \cdot e \\
&\leq (1+\mathbf{u})\left\{(1+\mathbf{u})^n \mathrm{fl}\left(|R|\left(t + r_{\mathrm{rad}}\right)\right) + n\underline{\mathbf{u}} \cdot e\right\} + n\underline{\mathbf{u}} \cdot e \\
&\leq (1+\mathbf{u})^{n+1} \mathrm{fl}\left(|R|\left(t + r_{\mathrm{rad}}\right)\right) + n(2+\mathbf{u})\underline{\mathbf{u}} \cdot e \\
&\leq (1+\mathbf{u})^{n+1} \mathrm{fl}\left(|R|\left(t + r_{\mathrm{rad}}\right)\right) + 2(1+\mathbf{u})\mathbf{u}_{\mathrm{N}} \cdot e \\
&\leq (1+\mathbf{u})^{n+2} \mathrm{fl}\left(|R|\left(t + r_{\mathrm{rad}}\right) + 2\mathbf{u}_{\mathrm{N}} \cdot e\right) \\
&\leq \mathrm{fl}\left(\frac{|R|\left(t + r_{\mathrm{rad}}\right) + 2\mathbf{u}_{\mathrm{N}} \cdot e}{1-(n+3)\mathbf{u}}\right) =: q.
\end{aligned}$$

Inserting this into (4.8),

$$|R(A\widetilde{x}-b)| \leq \mathrm{fl}\left(|R \cdot r_{\mathrm{mid}}|\right) + q \leq (1+\mathbf{u})\mathrm{fl}\left(|R \cdot r_{\mathrm{mid}}| + q\right).$$

So from (2.8) and (2.12), we obtain the following theorem:

THEOREM 4.3. *Let A, b, $\widetilde{x}$ and R be as in Theorem 3.1. Let $r_{\mathrm{mid}}, r_{\mathrm{rad}} \in \mathbb{F}^n$ satisfy*

$$r_{\mathrm{mid}} - r_{\mathrm{rad}} \leq A\widetilde{x} - b \leq r_{\mathrm{mid}} + r_{\mathrm{rad}}$$

and $r_{\mathrm{rad}} \geq \mathbf{0}$. Let t and q be defined by

$$t := \mathrm{fl}\left(\widetilde{\gamma}_{n+1} \max(|r_{\mathrm{mid}}|, \mathbf{u}_{\mathrm{N}} \cdot e)\right) \quad \textit{and} \quad q := \mathrm{fl}\left(\frac{|R|\left(t + r_{\mathrm{rad}}\right) + 2\mathbf{u}_{\mathrm{N}} \cdot e}{1-(n+3)\mathbf{u}}\right),$$

respectively. Then the following inequality holds in the presence of underflow:

$$\|R(A\widetilde{x}-b)\|_\infty \leq \mathrm{fl}\left(\frac{\left\||R \cdot r_{\mathrm{mid}}| + q\right\|_\infty}{1-2\mathbf{u}}\right). \tag{4.10}$$

It still remains the problem how to calculate r_{mid} and r_{rad}. Using (2.18), we can obtain

$$r_1 - r_2 \leq A\widetilde{x} - b \leq r_1 + r_2,$$

where

$$r_1 := \mathrm{fl}\,(A\widetilde{x} - b) \quad \text{and} \quad r_2 := \zeta_{n+1}(|A|\,|\widetilde{x}| + |b|) + n\underline{\mathbf{u}} \cdot e. \tag{4.11}$$

Using (2.3) and (2.24),

$$|A|\,|\widetilde{x}| + |b| \le (1+\mathbf{u})^{n+1}\mathrm{fl}\,(|A|\,|\widetilde{x}| + |b|) + n\underline{\mathbf{u}} \cdot e.$$

It follows that

$$\begin{aligned} r_2 &\le \zeta_{n+1}\{(1+\mathbf{u})^{n+1}\mathrm{fl}\,(|A|\,|\widetilde{x}| + |b|) + n\underline{\mathbf{u}} \cdot e\} + n\underline{\mathbf{u}} \cdot e \\ &\le \zeta_{n+1}(1+\mathbf{u})^{n+1}\mathrm{fl}\,(|A|\,|\widetilde{x}| + |b|) + n(1+\zeta_{n+1})\underline{\mathbf{u}} \cdot e \\ &\le \zeta_{2n+2}\mathrm{fl}\,(|A|\,|\widetilde{x}| + |b|) + \zeta_{2n+2}\mathbf{u}^{-1}\mathbf{u}_{\mathrm{N}} \cdot e \\ &\le \zeta_{2n+3}\mathrm{fl}\,\left((|A|\,|\widetilde{x}| + |b|) + \mathbf{u}^{-1}\mathbf{u}_{\mathrm{N}} \cdot e\right) \\ &\le \mathrm{fl}\left(\widetilde{\gamma}_{2n+4}\left\{(|A|\,|\widetilde{x}| + |b|) + \mathbf{u}^{-1}\mathbf{u}_{\mathrm{N}} \cdot e\right\}\right) =: r_3 \in \mathbb{F}^n. \end{aligned}$$

Here, we used $n(1+\zeta_{n+1})\underline{\mathbf{u}} \le \zeta_{2n+2}\mathbf{u}^{-1}\mathbf{u}_{\mathrm{N}}$. Thus, we can use Theorem 4.3 by setting r_1 and r_3 to r_{mid} and r_{rad}, respectively.

REMARK 4.4. *The calculation of residual $A\widetilde{x} - b$ usually causes big cancellation. This means that the radius r_{rad} may be much larger than the absolute value of the midpoint r_{mid} when using usual floating-point arithmetic. In such case, the quality of β calculated by the algorithms in this paper may be poor. Provided we can prove $\|RA - I\|_\infty \le \alpha < 1$, the quality of our error bounds is determined by the quality of the bound β. Thus any algorithm for a more accurate computation of the residual $A\widetilde{x} - b$ (cf., e.g. [8]) will improve our error bounds.*

We can now present an algorithm for calculating upper bounds on α and β by floating-point arithmetic only in round-to-nearest.

ALGORITHM 4.5. *Calculation of α and β such that $\|RA - I\|_\infty \le \alpha$ and $\|R(A\widetilde{x} - b)\|_\infty \le \beta$ without directed rounding.*

function $[\alpha, \beta] = \mathrm{std}(A, b, \widetilde{x})$
  if $(3n+2)\mathbf{u} \ge 1$, error('`verification failed.`'), end
  Compute a preconditioner R by a favorite algorithm
  $\alpha_1 = \mathrm{fl}\,(\|RA - I\|_\infty)$;
  if $\alpha_1 \ge 1$, error('`verification failed.`'), end
  $\alpha_2 = \mathrm{fl}\,(\||R|\,(|A|\,e)\|_\infty)$;
  $\alpha = \mathrm{fl}\left(\dfrac{\alpha_1 + \mathrm{fl}\,(\widetilde{\gamma}_{3n+2}(\alpha_2 + 2))}{1 - 2\mathbf{u}}\right)$;
  $r_{\mathrm{mid}} = \mathrm{fl}\,(A\widetilde{x} - b)$;
  $r_{\mathrm{rad}} = \mathrm{fl}\left(\widetilde{\gamma}_{2n+4}\left\{(|A|\,|\widetilde{x}| + |b|) + \mathbf{u}^{-1}\mathbf{u}_{\mathrm{N}} \cdot e\right\}\right)$;
  $t = \mathrm{fl}\,(\widetilde{\gamma}_{n+1}\max(|r_{\mathrm{mid}}|, \mathbf{u}_{\mathrm{N}} \cdot e))$;
  $q = \mathrm{fl}\left(\dfrac{|R|\,(t + r_{\mathrm{rad}}) + 2\mathbf{u}_{\mathrm{N}} \cdot e}{1 - (n+3)\mathbf{u}}\right)$;
  $\beta = \mathrm{fl}\left(\dfrac{\||R \cdot r_{\mathrm{mid}}| + q\|_\infty}{1 - 2\mathbf{u}}\right)$;

Algorithm 4.5 requires $\frac{10}{3}n^3$ flops in total, so that the theoretical ratio of computational cost to the LU factorization is 5. Basically, it holds in Algorithm 4.5 that

$$\alpha \sim \alpha_1 + c_n\mathbf{u} \cdot \mathrm{cond}(A),$$

TABLE 5.1
Evaluation of an upper bound α on $\|RA - I\|_\infty$ for A being a 1000×1000 random matrix.

cond(A)	Alg. 4.1	Alg. 4.5
10^3	8.11e-11	1.86e-08
10^5	5.45e-09	1.31e-06
10^7	3.78e-07	9.23e-05
10^9	3.43e-05	8.49e-03
10^{11}	2.81e-03	6.52e-01
10^{13}	2.25e-01	*
10^{15}	*	*

$*$ verification failed ($\alpha \geq 1$)

TABLE 5.2
Evaluation of an upper bound β for $\|R(A\widetilde{x} - b)\|_\infty$ for A being a 1000×1000 random matrix.

cond(A)	Alg. 4.1	Alg. 4.5
10^3	1.68e-12	1.28e-08
10^5	1.11e-10	8.94e-07
10^7	7.50e-09	6.33e-05
10^9	7.11e-07	5.83e-03
10^{11}	5.91e-05	4.48e-01
10^{13}	4.95e-03	3.75e+01
10^{15}	4.26e-01	3.52e+03

where c_n is a constant related to n and $\mathrm{cond}(A) := \|A\|_2\|A^{-1}\|_2$. If $\alpha_1 \geq 1$, then the computed R is essentially not good preconditioner for A. Now we assume that $\alpha_1 < 1$, then Algorithm 4.5 can be applied to the problem with condition number cond(A) satisfying

$$\mathrm{cond}(A) \lesssim (c_n \mathbf{u})^{-1}.$$

This restriction is not avoidable as long as a priori error estimates such as Lemma 2.6 are used.

We stress that only round-to-nearest is used in the verification process by Algorithm 4.5. Thus we can obtain α and β without directed rounding.

5. Numerical results. We shall present the results of numerical experiments showing properties and limits of the proposed verification methods. All computations in this section are done by Intel Pentium IV 3.46GHz CPU and Matlab 7.0.4 with IEEE 754 double precision arithmetic.

First, we evaluate the accuracy and the limits of the proposed verification methods. Let A be a 1000×1000 matrix whose elements are floating-point random numbers distributed in $[-1, 1]$. Here, the matrix is constructed by the Higham's test matrix `randsvd` for floating-point matrices with specified condition number [3]. Therefore, the condition number of A can be controlled, so that we vary it from 10^3 to 10^{15}. Let $b := \mathrm{fl}(A \cdot e)$ for $e = (1, \ldots, 1)^T \in \mathbb{R}^n$. This means that the residual $Ae - b$ can be expected to be small, and the exact solution $A^{-1}b$ is corrupted by rounding errors in the computation of b, but satisfies $\|e - A^{-1}b\| \leq \mathcal{O}(\mathbf{u})\mathrm{cond}(A)$. In Tables 5.1, 5.2 and 5.2, the results of upper bounds α on $\|RA - I\|_\infty$, β on $\|R(A\widetilde{x} - b)\|_\infty$ and on $\left\|\widetilde{x} - A^{-1}b\right\|_\infty$ by the standard verification algorithm with directed rounding (Algorithm 4.1) and the proposed verification algorithm (Algorithm 4.5) are displayed. Note that our algorithms verify nonsingularity of A if error bounds are computed (i.e. $\alpha < 1$).

Table 5.1 shows that Algorithm 4.1 is more stable than Algorithm 4.5. It seems to be near the best possible result if we can use the directed rounding and if we do not use higher precision or similar techniques.

TABLE 5.3
Evaluation of an upper bound on $\left\|\widetilde{x} - A^{-1}b\right\|_\infty$ *for* A *being a* 1000×1000 *random matrix.*

cond(A)	Alg. 4.1	Alg. 4.5
10^3	1.68e-12	1.28e-08
10^5	1.11e-10	8.94e-07
10^7	7.50e-09	6.33e-05
10^9	7.11e-07	5.88e-03
10^{11}	5.93e-05	1.29e+01
10^{13}	6.39e-03	*
10^{15}	*	*

$*$ verification failed ($\alpha \geq 1$)

TABLE 5.4
Elapsed time [sec] for LU factorization and the proposed methods and their ratio.

n	LU	Alg. 4.1	Alg. 4.5
500	0.14	0.99 (7.0)	0.71 (5.0)
1000	0.95	7.1 (7.5)	4.99 (5.3)
1500	3.0	23.1 (7.7)	16.0 (5.3)
2000	6.9	54.1 (7.9)	36.9 (5.4)
2500	13.1	105 (8.0)	71.1 (5.4)
3000	22.3	180 (8.1)	122 (5.5)
3500	34.9	287 (8.2)	193 (5.5)
4000	51.6	434 (8.4)	290 (5.6)

Algorithm 4.5 is applicable to condition numbers up to 10^{11}.

Table 5.3 shows that Algorithm 4.1 is more accurate than Algorithm 4.5. The results of Algorithm 4.5 are worse by about factor 10^4 compared to these of Algorithm 4.1.

Next, we measure the speed of the proposed verification method. In Table 5.4, the elapsed time for Algorithms 4.1 and 4.5 are displayed. For comparison, the elapsed time for LU factorization is displayed, and its ratio for each verification algorithms are also displayed in parentheses.

Table 5.4 shows that the actual (measured) computational cost for the verification algorithms is comparable to that for LU factorization. Their ratios for LU factorization almost correspond to theoretical values.

6. Conclusions. We proposed a fast self-validating algorithm using only rounding to nearest for verifying the nonsingularity of coefficient matrices and calculating rigorous error bounds for an approximate solution of linear systems. Using the proposed method, we can not only achieve better portability of programs but can also implement them with every compiler and on every platform provided that round-to-nearest follows the IEEE 754 standard. Numerical results illustrated that the proposed method is less stable than previous verification algorithms with directed rounding. However, it is applicable to ill-conditioned problems to a certain extent. Finally, we stress that we can also apply the technique in this paper to faster verification algorithms by using a priori error estimates in [9]. Then, we can develop an algorithm performing verification in the same computing time as the one for standard Gaussian elimination.

Appendix A. Proofs of Lemma 2.3. In the following, we prove Lemma 2.3.

Proof of (2.8) For the first inequality, it holds that

$$\frac{1}{1-\mathbf{u}} - (1+\mathbf{u}) = \frac{1-(1+\mathbf{u})(1-\mathbf{u})}{1-\mathbf{u}} = \frac{\mathbf{u}^2}{1-\mathbf{u}} > 0,$$

which implies

$$(1+\mathbf{u})^n < \frac{1}{(1-\mathbf{u})^n}.$$

The proof of the second inequality is by induction. Obviously, the equality holds for $n = 1$. If it holds for $n = k$ that

$$\frac{1}{(1-\mathbf{u})^k} < \frac{1}{1-k\mathbf{u}},$$

then

$$\frac{1}{(1-\mathbf{u})^{k+1}} = \frac{1}{(1-\mathbf{u})(1-\mathbf{u})^k} < \frac{1}{1-\mathbf{u}} \cdot \frac{1}{1-k\mathbf{u}} < \frac{1}{1-(k+1)\mathbf{u}}. \quad \square$$

Proof of (2.9) From the definition of ζ_n, it follows that

$$(1+\mathbf{u})^p \zeta_n = (1+\mathbf{u})^p((1+\mathbf{u})^n - 1) \leq (1+\mathbf{u})^{n+p} - 1 = \zeta_{n+p}. \quad \square$$

Proof of (2.10)

$$\begin{aligned}\zeta_{n+p} - (\zeta_n + \zeta_p) &= (1+\mathbf{u})^{n+p} - (1+\mathbf{u})^n - \zeta_p \\ &= (1+\mathbf{u})^n((1+\mathbf{u})^p - 1) - \zeta_p = \zeta_n\zeta_p \geq 0. \quad \square\end{aligned}$$

Proof of (2.11) We prove $\zeta_n \leq \gamma_{n-1}$ because $\gamma_{n-1} \leq \widetilde{\gamma}_n$. The proof is by induction. Obviously, the equality holds for $n = 1$. If it holds for $n = k$ that $\zeta_k \leq \gamma_{k-1}$, then $(1+\mathbf{u})\zeta_k \leq (1+\mathbf{u})\gamma_{k-1}$ and

$$(1+\mathbf{u})^{k+1} - (1+\mathbf{u}) \leq (1+\mathbf{u})\frac{(k-1)\mathbf{u}}{1-(k-1)\mathbf{u}}.$$

It follows that

$$\zeta_{k+1} = (1+\mathbf{u})^{k+1} - 1 \leq \frac{(k-1)\mathbf{u}}{1-k\mathbf{u}} + \mathbf{u} \leq \frac{k\mathbf{u} - k\mathbf{u}^2}{1-k\mathbf{u}} \leq \gamma_k. \quad \square$$

Proof of (2.12)

$$\frac{1}{1-n\mathbf{u}}|a| = (1+\mathbf{u})^{-1}(1+\mathbf{u})\frac{|a|}{1-n\mathbf{u}} \leq (1+\mathbf{u})^{-1}\frac{|a|}{1-(n+1)\mathbf{u}} \leq \mathrm{fl}\left(\frac{\max(|a|, \mathbf{u}_{\mathrm{N}})}{1-(n+1)\mathbf{u}}\right). \quad \square$$

Proof of (2.13)

$$\zeta_n|a| = (1+\mathbf{u})^{-1}(1+\mathbf{u})\zeta_n|a| \leq (1+\mathbf{u})^{-1}\zeta_{n+1}|a| \leq \mathrm{fl}\left(\widetilde{\gamma}_{n+1}\max(|a|, \mathbf{u}_{\mathrm{N}})\right). \quad \square$$

REFERENCES

[1] ANSI/IEEE, *IEEE Standard for Binary Floating Point Arithmetic*, Std 754–1985 edition, IEEE, New York, 1985.

[2] G. I. Hargreaves, *Interval Analysis in MATLAB*, Numerical Analysis Report No. 416, Manchester Centre for Computational Mathematics, 2002.

[3] N. J. Higham, *Accuracy and Stability of Numerical Algorithms*, 2nd ed., SIAM Publications, Philadelphia, 2002.

[4] W. Kahan, J. D. Darcy, *How Java's Floating-Point Hurts Everyone Everywhere*, manuscript, 2001. `http://www.cs.berkeley.edu/~wkahan/JAVAhurt.pdf`

[5] R. B. Kearfott, M. Dawande, K. Du, C. Hu, *Algorithm 737; INTLIB: a portable Fortran 77 interval standard-function library*, ACM Trans. Math. Softw., 20:4 (1994), pp. 447–459.

[6] R. Klatte, U. Kulisch, A. Wiethoff, C. Lawo, M. Rauch, *C-XSC – A C++ Class Library for Extended Scientific Computing*, Springer-Verlag, Heidelberg, 1993.

[7] O. Knüppel, *PROFIL/BIAS – A fast interval library*, Computing, 53 (1994), pp. 277-287.

[8] T. Ogita, S. M. Rump, S. Oishi, *Accurate Sum and Dot Product*, SIAM J. Sci. Comput., 26:6 (2005), pp. 1955–1988.

[9] S. Oishi, S. M. Rump, *Fast verification of solutions of matrix equations*, Numer. Math., 90:4 (2002), pp. 755–773.

[10] S. M. Rump, *INTLAB – INTerval LABoratory*, Developments in Reliable Computing (Tibor Csendes ed.), Kluwer Academic Publishers, Dordrecht, 1999, pp. 77–104. `http://www.ti3.tu-harburg.de/~rump/intlab/`

[11] S. M. Rump, *Verification methods for dense and sparse systems of equations*, Topics in Validated Computations – Studies in Computational Mathematics (J. Herzberger ed.), Elsevier, Amsterdam, pp. 63–136, 1994.

[12] The MathWorks Inc., *MATLAB Users Guide, Version 7*, 2004.

[13] W. V. Walter, *ACRITH-XSC: A Fortran-like language for verified scientific computing*, Scientific Computing with Automatic Result Verification (E. Adams and U. Kulisch eds.), Academic Press, New York, pp. 45–70, 1993.

Appeared as a Technical Report on the home page of the Advanced Research Institute of Science and Engineering, Waseda University, on July 26, 2005.