Journal of Computational and Applied Mathematics 242 (2013) 157–184

Contents lists available at SciVerse ScienceDirect

Journal of Computational and Applied Mathematics

journal homepage: www.elsevier.com/locate/cam

Accurate solution of dense linear systems, part I: Algorithms in rounding to nearest

Siegfried M. Rump *

Institute for Reliable Computing, Hamburg University of Technology, Schwarzenbergstraße 95, Hamburg 21071, Germany
Waseda University, Faculty of Science and Engineering, 3–4–1 Okubo, Shinjuku-ku, Tokyo 169–8555, Japan

ARTICLE INFO

Article history:
Received 12 August 2011
Received in revised form 22 August 2012

Keywords:
Linear system
Rounding to nearest
(extremely) ill-conditioned
Error-free transformation
Error analysis
Rigorous error bounds

ABSTRACT

We investigate how extra-precise accumulation of dot products can be used to solve ill-conditioned linear systems accurately. For a given p-bit working precision, extra-precise evaluation of a dot product means that the products and summation are executed in $2p$-bit precision, and that the final result is rounded into the p-bit working precision. Denote by $\mathbf{u} = 2^{-p}$ the relative rounding error unit in a given working precision. We treat two types of matrices: first up to condition number $\mathbf{u}^{-1}$, and second up to condition number $\mathbf{u}^{-2}$. For both types of matrices we present two types of methods: first for calculating an approximate solution, and second for calculating rigorous error bounds for the solution together with the proof of non-singularity of the matrix of the linear system. In the first part of this paper we present algorithms using only rounding to nearest, in Part II we use directed rounding to obtain better results. All algorithms are given in executable Matlab code and are available from my homepage.

1. Introduction and notation

The solution of a linear system $Ax = b$ is a ubiquitous task in numerical computations. In Part I and II of this paper we present different methods to compute guaranteed error bounds for the solution of a linear system, i.e. with a *certified* accuracy. The methods in this Part I are based on norm estimates, in particular verifying convergence of some residual matrix by approximating its Perron vector, whereas the methods in Part II are based on the verification of the H-property of some matrix. Moreover, in the present Part I of the paper we (1) present a method to compute an approximation for *extremely* ill-conditioned linear systems which is *likely* to be accurate.

In the present Part I all algorithms use only the four basic floating-point operations in rounding to nearest, in Part II directed rounding is used as well. The challenge for the first part is to use only standard Matlab code in rounding to nearest without additional mex-files and to derive simple and fast algorithms. All algorithms in both parts are presented in executable Matlab-code.

Let a floating-point format with relative rounding error unit $\mathbf{u}$ be given. The forward error of an approximation $\widetilde{x}$ computed by a standard algorithm like Gaussian elimination is of the order $\mathbf{u} \cdot \mathrm{cond}(A)$ [1]. This naturally bounds the applicability to matrices A with $\mathrm{cond}(A) \lesssim \mathbf{u}^{-1}$, which means $\mathrm{cond}(A) \lesssim 10^{16}$ in IEEE 754 double precision (binary64). For larger condition numbers, $\widetilde{x}$ is expected to have no correct digit.

Skeel [2] showed that one step of the classical residual iteration with the residual computed in *working precision* produces a backward stable result. It is also known that, for condition numbers up to $\mathrm{cond}(A) \lesssim \mathbf{u}^{-1}$, an almost maximally accurate

* Correspondence to: Institute for Reliable Computing, Hamburg University of Technology, Schwarzenbergstraße 95, Hamburg 21071, Germany.
E-mail address: rump@tu-harburg.de.

0377-0427/$ – see front matter
doi:10.1016/j.cam.2012.10.010

Table 1.1
Constants for single (binary32) and double (binary64) precision in the IEEE 754 floating-point standard.

	p	$e_{\min}$	$e_{\max}$
Single precision	24	−126	127
Double precision	53	−1022	1023

result (of order $\mathbf{u}$) is achieved if the dot products in the residual iteration are accumulated in twice the working precision (see also [3]). But for condition numbers of the order $\mathbf{u}^{-1}$ and larger, again no correct digit can be expected. So basically we face the dichotomy of either high accuracy or no accuracy at all.

We will show that the accumulation of dot products in twice the working precision (with result rounded into working precision) suffices to compute an accurate approximation of the solution of a linear system for condition numbers up to $\mathrm{cond}(A) \lesssim \mathbf{u}^{-2}$. Moreover, we show how rigorous error bounds can be computed including the proof on non-singularity of the input matrix A. Practical experience suggests that this approach works successfully up to condition numbers $c \cdot \mathrm{cond}(A) \lesssim \mathbf{u}^{-2}$ with $c \approx n^2$ in IEEE 754 binary64 (double precision). This factor will be improved to about $c \approx n$ in Part II of this paper.

We want to stress that our bounds are mathematically completely rigorous including all possible sources of errors (provided the compiler and operating system work to their specification). Although it is in principle known how to compute rigorous error bounds in rounding to nearest, the corresponding algorithms are involved, and taking care of underflow they become unwieldy. One reason to divide the paper in two parts is to clearly distinguish between algorithms using solely rounding to nearest (Part I), and those using directed rounding (Part II).

There are other, very good but not completely rigorous approaches. For example, an upper bound of the condition number $\mathrm{cond}(A)$ implies an error bound of an approximate solution of $Ax = b$. There are many $\mathcal{O}(n^2)$ condition number estimators (cf. [4,1]), usually providing good approximations. By the principle of the methods these are *lower bounds* for the condition number, and for every estimator counterexamples are known where the condition number is grossly underestimated.

In another approach [5,6] the authors use a statistic way for estimating rounding errors. Using a so-called stochastic arithmetic they propose a method to determine the number of significant digits of a computed result. Also those results are correct with a high degree of certainty, but not with complete rigor.

Yet another approach [3] uses the vast experience in solving linear systems very thoughtfully to produce approximations with "likely correct error terms" [3]. It seems that no counterexample is known where the claimed accuracy is not valid, but it is not *proved* to be correct.

To repeat it, beyond accurate approximations for very ill-conditioned linear systems, we are also interested in mathematically rigorous error bounds. Such rigorous bounds are, for example, mandatory in so-called "computer-assisted proofs" [7], which recently gain interest. For example, Tucker [8] received the 2004 EMS prize awarded by the European Mathematical Society for "giving a rigorous proof that the Lorenz attractor exists for the parameter values provided by Lorenz. This was a long standing challenge to the dynamical system community, and was included by Smale in his list of problems for the new millennium. The proof uses computer estimates with rigorous bounds based on higher dimensional interval arithmetics".

Concerning notation denote by $\mathbb{F}$ a set of p-bit binary floating-point numbers including ∞ and NaN, i.e. [9]

$$\mathbb{F} = \{M \cdot 2^{e-p+1} \mid M, e \in \mathbb{Z},\ |M| \le 2^p - 1,\ e_{\min} \le e \le e_{\max}\} \cup \{-\infty, +\infty, \mathrm{NaN}\}. \tag{1.1}$$

For single (binary32) and double (binary64) precision in the IEEE 754 floating-point standard [10,11] the constants are as in Table 1.1. We assume floating-point operations in rounding to nearest, tie to even, as in the IEEE 754 standard. That means there is a mapping $\mathrm{fl} : \mathbb{R} \to \mathbb{F}$ such that $|\mathrm{fl}(x) - x| = \min_{f \in \mathbb{F}} |f - x|$ for all $x \in \mathbb{R}$, and for $a, b \in \mathbb{F}$ floating-point operations $\circ_{\mathrm{fl}} : \mathbb{F} \times \mathbb{F} \to \mathbb{F}$ with $\circ \in \{+, -, \cdot, /\}$ are defined by

$$a \circ_{\mathrm{fl}} b := \mathrm{fl}(a \circ b). \tag{1.2}$$

Therefore the result $a \circ_{\mathrm{fl}} b \in \mathbb{F}$ is a best approximation of $a \circ b \in \mathbb{R}$. The relative rounding error unit is defined by $\mathbf{u} = 2^{-p}$. A floating-point number $M \cdot 2^{e-p+1}$ is normalized if $|M| \ge 2^{p-1}$, the smallest normalized positive floating-point number is `realmin` $= 2^{e_{\min}}$, and the smallest unnormalized positive floating-point number is `eta` $= 2^{e_{\min}-p+1}$. All algorithms are given in executable Matlab code using IEEE 754 double precision, but the results are valid in any floating-point arithmetic complying with the IEEE 754 standard.

Comparison between vectors and matrices is always to be understood entrywise, for example $x \le y$ for $x, y \in \mathbb{R}^n$ means $x_i \le y_i$ for $1 \le i \le n$. Executable Matlab-code is written using the "verbatim"-font. For instance, `C=A*B` means that `C` is the result of the floating-point multiplication `A*B`, where `A` and `B` are compatible quantities (scalar, vector, matrix). For analyzing the error we use ordinary mathematical notation, for example in $P = \mathtt{A} \cdot \mathtt{B}$ the verbatim-font is used for floating-point quantities so that P is the exact (real) product of `A` and `B`. For $\mathtt{A}, \mathtt{B} \in \mathbb{F}^{n \times n}$ this implies $|P - \mathtt{C}| \sim \mathbf{u}|\mathtt{A}| \cdot |\mathtt{B}|$.