悪条件連立一次方程式の精度保証付き数値計算法

太田 貴久 * 荻田 武史 †,* S. M. Rump‡ 大石 進一 *
* 早稲田大学理工学術院 † 科学技術振興機構 ‡ Hamburg-Harburg 工科大学

Numerical Verification Method for Arbitrarily Ill-conditioned Linear Systems

Takahisa Ohta* Takeshi Ogita†,* Siegfried M. Rump‡ Shin'ichi Oishi
* Waseda University † Japan Science and Technology Agency
‡ Technical University Hamburg-Harburg

Abstract. This paper is concerned with the problem of verifying an accuracy of a numerical solution of a linear system with an arbitrarily ill-conditioned coefficient matrix. In this paper, a method of obtaining an accurate numerical solution of such a linear system and its verified error bound is proposed. The proposed method is based on the accurate computation of dot product and IEEE standard 754 arithmetic. A verified and accurate numerical solution with a desired tolerance can be obtained by the proposed method with iterative refinement. Numerical results are presented for illustrating the effectiveness of the proposed method.

1 概要

本論文では，連立一次方程式

$$Ax = b \tag{1}$$

の数値解 $\tilde{x}$ の精度保証付き数値計算法について考える．ただし，A は実 $n \times n$ 行列で，b は実 n 次元ベクトルとする．ここで，数値解とは計算機によって得られる近似解のことであり，精度保証とは厳密解に対する近似解の定量的誤差評価のことを指す．

式 (1) に対し

$$\kappa(A) := \|A\| \cdot \|A^{-1}\|$$

をその条件数という．式 (1) の厳密解を $x^* := A^{-1}b$ とすると (ただし，A が正則のとき)，b が $b + \Delta b$ に変化したとき，解は $\kappa(A)\|\Delta b\|$ のオーダで変化する．すなわち，解の変化量を Δx として連立一次方程式 $A(x + \Delta x) = b + \Delta b$ を考えると

$$\frac{\|\Delta x\|}{\|x^*\|} \leq \kappa(A)\frac{\|\Delta b\|}{\|b\|} \tag{2}$$

が成り立つ．したがって，条件数の大きな問題では，右辺ベクトルの少しの変化が解の大きな変化を引き起こすことがわかる (条件数についてのより多くの考察については文献 [2] を参照されたい)．IEEE754 規格の倍精度浮動小数点数演算で計算する場合，仮数部の相対精度は高々$2^{-53} = 1.11\cdots \times 10^{-16}$ であるから，条件数が 10^{16} 程度の大きさになると，丸め誤差の混入する数値計算での数値解の精度は仮数部で 1 桁あるかないかというオーダになり，倍精度演算の限界となる．ここでは，10^{16} 程度より大きな条件数を持つ問題を悪条件な問題と呼んでいる．

本論文では，条件数の非常に大きい行列に対して，その逆行列を高精度に求める方法やそれを応用した連立一次方程式の数値解の精度保証法について検討している．倍精度演算の範囲で取り扱うことができない問題では高精度演算が不可欠であるが，すべての計算に対して多倍長演算などを適用するのは計算コスト的に有効ではない．一方で，内積計算や行列積など特定の計算に限れば，それらの高精度演算について比較的高速なアルゴリズムが利用可能である (たとえば，文献 [1, 3, 4, 6])．そこで，本論文では，与えられた行列の条件数が非常に大きい場合でも，内積計算 (行列・ベクトル積や行列積を含む) さえ高精度に実行できれば，その他には多倍長演算などを必要としない高速な精度保証法を提案する．また，数値例によりその有効性を示す．

2　浮動小数点演算と高精度な内積計算

2.1　準備

行列 $A = (a_{ij})$ が実とは，その要素 a_{ij} が全て実数であることをいう．以下，本論文では，記号として，$\mathbb{R}$ で実数の集合，$\mathbb{R}^n$ で実 n 次元ベクトル空間を表す．また，ベクトル $x \in \mathbb{R}^n$ の最大値ノルムを

$$\|x\|_\infty := \max_{1\le i\le n} |x_i| \tag{3}$$

で，行列 $A \in \mathbb{R}^{n\times n}$ の (最大値ノルムによる) 作用素ノルムを

$$\|A\|_\infty := \max_{1\le i\le n} \sum_{j=1}^{n} |a_{ij}| \tag{4}$$

で表す．また，ベクトル p, q に対して $|p| = (|p_i|)$ で $p \le q$ はすべての i に対して $p_i \le q_i$ であることを意味する．行列についても同様の表記を用いる．

2.2　IEEE754 の倍精度浮動小数点数規格

本論文では，連立一次方程式 $Ax = b$ の係数行列 A と右辺ベクトル b の各要素は IEEE754 規格に従う浮動小数点数とし，解の計算には浮動小数点数演算を用いるという立場を取る．IEEE754 規格の中では，32 ビットの単精度浮動小数点数と 64 ビットの倍精度浮動小数点数及びそれらの演算が標準化されているが，簡単のため，以下では倍精度浮動小数点数での計算のみを考えることにする．

まず，IEEE754 規格を満たす倍精度浮動小数点数に成り立つ重要な性質を概観する．この IEEE754 規格は数学的に厳密な性質を備えたもので，精度保証付き数値計算に必要な有向丸めの概念も実装されている．これを利用して，最近，Oishi と Rump [5] は，内積計算や行列積の精度保証が 2 回の丸めのモードの変更でできることを指摘し，それを基礎に丸めモード制御に基づく精度保証付き数値計算法を開発した．

以下，$\mathbb{F}$ で IEEE754 規格の倍精度浮動小数点数の集合を表すことにする． IEEE754 規格では倍精度浮動小数点数演算は丸めのモードを指定することにより定義されている．丸めのモードは IEEE754 では 4 種類あるが，ここでは，本論文に関係する 3 つの丸めのモードの定義を紹介する．

(1) 下への丸め ($-\infty$ 方向への丸め) $c \in \mathbb{R}$ を $f \le c$ を満たす最も大きな浮動小数点数 $f \in \mathbb{F}$ へ丸める．下への丸めを $\nabla : \mathbb{R} \to \mathbb{F}$ と表す．

(2) 上への丸め ($+\infty$ 方向への丸め) $c \in \mathbb{R}$ を $f \ge c$ を満たす最も小さな浮動小数点数 $f \in \mathbb{F}$ へ丸める．上への丸めを $\triangle : \mathbb{R} \to \mathbb{F}$ と表す．

(3) 最近点への丸め $c \in \mathbb{R}$ を $|f - c|$ が最小となる浮動小数点数 $f \in \mathbb{F}$ へ丸める．これを $\square : \mathbb{R} \to \mathbb{F}$ と表す．

IEEE754 規格での倍精度浮動小数点数演算は $\circ \in \{+, -, \cdot, /\}$ と $\bigcirc \in \{\triangle,\ \nabla,\ \square\}$ に対して

$$fl_{\bigcirc}(x \circ y) = \bigcirc(x \circ y), \quad \forall x, y \in \mathbb{F} \tag{5}$$

が満たされるように定義される．ここで，式 (5) の左辺の $fl_{\bigcirc}$ は丸めモードが $\bigcirc$ である浮動小数点数演算を表し，$x \circ y$ は通常の実数演算を表すものとする．

この定義から任意の $x, y \in \mathbb{F}$ と $\circ \in \{+, -, \times, /\}$ に対して

$$fl_{\nabla}(x \circ y) \le x \circ y \le fl_{\triangle}(x \circ y)$$

が成り立つ．したがって，特に，$x_i, y_i \in \mathbb{F},\ (i = 1, 2, \ldots, n)$ のとき

$$fl_{\bigtriangledown}\left(\sum_{i=1}^{n} x_i \cdot y_i\right) \leq \sum_{i=1}^{n} x_i \cdot y_i \leq fl_{\bigtriangleup}\left(\sum_{i=1}^{n} x_i \cdot y_i\right)$$

が成り立つことがわかる．これは内積の上限と下限が厳密に計算できることを示している．また，倍精度浮動小数点数の絶対値は正しく計算でき，2 つの倍精度浮動小数点数の大小も正しく判定できる．したがって，$v \in \mathbb{R}^n$ 及び $\underline{v}, \overline{v} \in \mathbb{F}^n$ が

$$\underline{v}_i \leq v_i \leq \overline{v}_i$$

を満たしているとき，ベクトル v の最大値ノルムの上限は

$$\|v\|_\infty \leq \max_{1 \leq i \leq n} \left(\max\{|\underline{v}_i|, |\overline{v}_i|\}\right) \tag{6}$$

によって計算できることがわかる．同様に，$A = (a_{ij}) \in \mathbb{R}^{n \times n}$ 及び $\underline{A} = (\underline{a}_{ij}), \overline{A} = (\overline{a}_{ij}) \in \mathbb{F}^{n \times n}$ が

$$\underline{a}_{ij} \leq a_{ij} \leq \overline{a}_{ij}$$

を満たすとき，A の最大値ノルムの上限は

$$\|A\|_\infty \leq \max_{1 \leq i \leq n} \left[fl_{\bigtriangleup}\left(\sum_{j=1}^{n} \max\{|\underline{a}_{ij}|, |\overline{a}_{ij}|\}\right)\right] \tag{7}$$

によって計算できる．また，浮動小数点演算においてオーバーフローは起きないものとし[1]，アンダーフローは許容することにする．

以上の事柄を基礎に，文献 [5] では式 (1) の高速精度保証の理論が展開されており，それは本論文の議論の基礎ともなっている．本論文で用いるベクトルの内積や行列積の高精度演算についてはつぎの小節で紹介される．

2.3 高精度な内積計算

ベクトル $x, y \in \mathbb{F}^n$ に対して内積計算

$$x^T y = \sum_{i=1}^{n} x_i y_i$$

は科学技術計算において重要な役割を占めるため，高精度に内積を計算するための数々のアルゴリズムが開発されてきた (たとえば，文献 [1, 3, 4, 6])．一般的に，多倍長演算はハードウェア命令で実行できる倍精度演算に比べてかなり低速 (数十倍〜数百倍程度) であるが，内積の高精度演算に限ってみれば比較的高速 (数倍〜数十倍程度) に実行可能である．現在までに開発されてきた主な方法を以下に示す．

- Kulisch, Miranker による “long accumulator” を用いた完全精度内積演算 [1]
- Priest による “doubly compensated summation” [6, 2]
- XBLAS [3]

[1] 通常，オーバーフローが発生した場合，計算は中断される．

- Ogita, Rump, Oishi による高速な任意精度内積演算アルゴリズム [4]

詳細についてはそれぞれの文献を，その他のアルゴリズムについては，たとえば文献 [2] を参照されたい．

本論文で提案する連立一次方程式のための精度保証法は内積計算のアルゴリズムに依存しない．ここでは，完全精度内積演算のように正確に求めた内積の値を必要な精度だけ得ることができるという仮定のもとに議論を進める．

汎用的な関数として

$$\mathbf{DotExact}(expression, parameter)$$

は，条件 *parameter* に従って *expression* を高精度に計算するものとする．たとえば，$A \in \mathbb{F}^{m\times p}, B \in \mathbb{F}^{p\times n}$ に対して行列乗算 $A \cdot B$ を考えたとき

$$C_{1:k} = \mathbf{DotExact}(A \cdot B, k)$$

は

$$|C_i| \geq 2^{52} \cdot |C_{i+1}|, \quad i = 1, 2, \ldots, k-1$$

であり (つまり，仮数部がほとんど影響し合わない順番に並んでいる)，かつ

$$|\sum_{i=1}^{k} C_i - A \cdot B| \leq \max(2^{-52} \cdot |C_k|, realmin \cdot E) \tag{8}$$

となるような $C_i \in \mathbb{F}^{m\times n}, (i = 1, 2, \ldots, k)$ を求めるものとする．ただし，$realmin := 2^{-1022}$ は IEEE754 倍精度浮動小数点数で表現できる正規化数のうち正の最小のものであり，E はすべての要素が 1 の $m \times n$ 行列とする．これらは，アンダーフローが起きたときにも精度保証が厳密であるために必要となる．また

$$[C_{1:k}, C_{\mathrm{rad}}] = \mathbf{DotExact}(A \cdot B, \{\text{'midrad'}, k\})$$

は

$$|C_i| \geq 2^{52} \cdot |C_{i+1}|, \quad i = 1, 2, \ldots, k-1$$

かつ

$$|\sum_{i=1}^{k} C_i - A \cdot B| \leq C_{\mathrm{rad}}$$

となるような $C_i \in \mathbb{F}^{m\times n}, (i = 1, 2, \ldots, k)$ 及び $C_{\mathrm{rad}} \in \mathbb{F}^{m\times n}$ を求めるものとする．さらに

$$A_{1:k} := A_1 + A_2 + \ldots + A_k = \sum_{i=1}^{k} A_i, \quad A_i \in \mathbb{F}^{n\times n}$$

に対しても

$$\mathbf{DotExact}(A_{1:k} \cdot B, parameter)$$

のような使い方ができるものとする．これは

$$\mathbf{DotExact}(A_1B + A_2B + \ldots + A_kB, parameter)$$

と同値である．同様に

$$\mathbf{DotExact}(A_{1:k_1} \cdot B_{1:k_2}, parameter)$$

は

$$\mathbf{DotExact}(A_1B_1 + A_2B_1 + \ldots + A_{k_1}B_{k_2}, parameter)$$

を意味するものとする．

3 提案方式

3.1 成分毎の精度保証法

近年，Oishi と Rump [5] は式 (1) の数値解に対する高速精度保証法を提案した．数値解に対して成分毎に誤差評価を行うには次の Yamamoto の定理 [9] が有用である．解ベクトルの要素において，その絶対値の大きさにばらつきがあるときは，成分毎の誤差評価を行うことにより絶対値が相対的に小さい要素に対しても過大評価の少ない精度保証が可能となる．

定理 1 (Yamamoto [9]) A を $n \times n$ 実行列，b を n 次元実ベクトルとして連立一次方程式

$$Ax = b$$

を考える．適当な $n \times n$ 実行列 R が存在して[2]

$$\|RA - I\|_\infty < 1 \tag{9}$$

を満たすとき，A は正則で，任意の n 次元実ベクトル $\tilde{x}$ に対して[3]

$$|\tilde{x} - A^{-1}b| \leq |R(A\tilde{x} - b)| + \frac{\|R(A\tilde{x} - b)\|_\infty}{1 - \|RA - I\|_\infty} t \tag{10}$$

が成り立つ．ただし，t は $G := RA - I$ としたときに

$$(\sum_{j=1}^{n} |G_{1j}|, \sum_{j=1}^{n} |G_{2j}|, \ldots, \sum_{j=1}^{n} |G_{nj}|)^T \leq t \tag{11}$$

を満たす n 次元ベクトルであり，I は n 次元単位行列である． □

この定理では $\tilde{x}$ は任意であるが，我々の実装法においては，2.3 節で紹介した高精度内積計算アルゴリズム **DotExact** を用いた残差反復法により，必要な回数だけ反復した式 (1) の近似解を取るものとする．具体的には，式 (10) の右辺の分子に現れる $A\tilde{x} - b$ を **DotExact** を用いて高精度に計算する．R についても，**DotExact** を用いた反復計算により $\|RA - I\|_\infty < 1$ を満たすような R を求める．

提案する精度保証法の詳細については，次の小節で述べる．

3.2 条件数の大きい行列に対する精度保証法

A の条件数が 10^{16} 程度より大きいと，倍精度演算の範囲では $\|RA - I\|_\infty$ の値が 1 を超えてしまい，3.1 節で示した精度保証法が適用できなくなる．これは，計算中の丸め誤差の影響や，R の表現に必要な精度の不足によって，R が A の逆行列としての情報をほとんど持たないことを意味する．しかしながら，そのような場合においても，実は R は A の前処理としての情報を依然として含んでいることが知られている [8]．この性質を利用して

$$\|RA - I\|_\infty < 1$$

を満たすような R を高精度に求める方法を提案する．

---

[2] 実際の応用では R としては A の近似逆行列を取るのが普通である．

[3] やはり実際の応用では $Ax = b$ の近似解を $\tilde{x}$ とする．

まず，倍精度演算の性質をできるだけ有効に活用するために，R を

$$R = R_1 + R_2 + \ldots + R_k = \sum_{i=1}^{k} R_i, \quad R_i \in \mathbb{F}^{n\times n}$$

のように，要素が倍精度浮動小数点数である行列の和として表現することを考える．ただし

$$|R_i| \geq 2^{52} \cdot |R_{i+1}|, \quad i = 1, 2, \ldots, k-1$$

とする．ここで，我々は文献 [8] の方法を拡張した次のアルゴリズムを提案する．尚，本論文ではアルゴリズムを MATLAB に近い表記で記述する．

アルゴリズム 1 高精度に $R = \sum_{i=1}^{k} R_i$ を求めるアルゴリズム：

function $R_{1:k}^{(k)} = \mathbf{AccInv1}(A, k)$
$R_1^{(1)} = \mathbf{inv}(A)$ % A の逆行列の計算 (倍精度演算)
for $i = 2 : k$
  $C = \mathbf{DotExact}(R_{1:i-1}^{(i-1)} \cdot A, 1)$ % $C \leftarrow R \cdot A$ (高精度演算, 結果は倍精度)
  $T = \mathbf{inv}(C)$ % $T \approx C^{-1}$ (倍精度演算)
  $R_{1:i}^{(i)} = \mathbf{DotExact}(T \cdot R_{1:i-1}^{(i-1)}, i)$ % $R_{new} \leftarrow T \cdot R$ (高精度演算)
end

まず，$R^{(1)} := R_1$ を A の最初の前処理行列と考え，$R_1 A$ を高精度に計算し，結果を倍精度に丸めて

$$C \approx R_1 A$$

のようにする．次に，C の (近似) 逆行列を通常の倍精度演算で

$$T \approx C^{-1} \approx (R_1 A)^{-1}$$

のように求める．そして，TR_1 を高精度に計算し，新たに求めた R_1, R_2 を次の前処理行列 $R^{(2)} := R_1 + R_2$ とすると，結果は

$$R^{(2)} = R_1 + R_2 \approx TR_1 \approx (R_1 A)^{-1} R_1 = A^{-1}$$

となることが期待できる．同様に上記の操作を繰り返し，$R^{(k)} := R_1 + R_2 + \ldots + R_k = \sum_{i=1}^{k} R_i$ を求める．

このアルゴリズムにより，通常の倍精度演算と比較して高精度な R を計算することが可能となり，実際，条件数 $\kappa(A)$ が

$$\kappa(A) \lesssim 10^{16(k-1)} \sim 10^{16k} \tag{12}$$

程度の問題であれば A の正則性の保証が可能となる (これについては，後で数値実験によって確認する)．ただし，一般的には A の条件数は未知であるため，計算された R が $\|RA - I\|_\infty < 1$ を満たすかどうかは保証されない．すなわち，事前に R にはどれだけの精度が必要か (k をいくつにすれば良いか) はわからないため，A の性質によって $\|RA - I\|_\infty < 1$ を満たすような R を自動的に反復改良しながら計算できるアルゴリズムが望ましい．

そこで，反復毎に $\|RA - I\|_\infty < 1$ の判定をしながら R を更新していくアルゴリズムを以下に示す．また，定理 1 を適用するため式 (11) を満たす t を求める操作も加える．

アルゴリズム 2 $\|RA - I\|_\infty \leq \alpha < 1$ を満たす R, α 及び式 (11) を満たす t を求めるアルゴリズム：

function $[R_{1:k}, k, \alpha, t] = \mathbf{AccInv3}(A, \ell_{\max})$
$k = 1$
$R_1 = \mathbf{inv}(A)$
$\underline{G} = \mathit{fl}_{\triangledown}(RA - I)$ % $RA - I$ の下限
$\overline{G} = \mathit{fl}_{\triangle}(RA - I)$ % $RA - I$ の上限
$\overline{G} = \max(|\underline{G}|, |\overline{G}|)$ % $|RA - I|$ の上限
for $i = 1 : n$
  $t_i = \mathit{fl}_{\triangle}\left(\sum_{j=1}^{n} \overline{G}_{ij}\right)$ % $\sum_{j=1}^{n} \overline{G}_{ij} \le t_i$
end
$\alpha = \max_{1 \le i \le n} t_i$ % $\|RA - I\|_\infty$ の上限
if $\alpha < 1,$ return, end
while $k \le \ell_{\max}$ % $\ell_{\max}$: 最大反復回数
  $C = \mathbf{DotExact}(R_{1:k} \cdot A, 1)$
  if $\mathit{fl}_{\square}(\|C - I\|_\infty) < 10^{-3},$ break, end % 反復停止条件
  $T = \mathbf{inv}(C)$
  $R_{1:k+1} = \mathbf{DotExact}(T \cdot R_{1:k}, k + 1)$
  $k = k + 1$
end
$T = \mathit{fl}_{\square}(\max(2^{-52} \cdot |C|, \mathit{realmin} \cdot E))$ % $T_{ij} = \max(2^{-52} \cdot |C_{ij}|, \mathit{realmin})$
$\underline{G} = \mathit{fl}_{\triangledown}(C - I - T)$ % $RA - I$ の下限
$\overline{G} = \mathit{fl}_{\triangle}(C - I + T)$ % $RA - I$ の上限
$\overline{G} = \max(|\underline{G}|, |\overline{G}|)$ % $|RA - I|$ の上限
for $i = 1 : n$
  $t_i = \mathit{fl}_{\triangle}\left(\sum_{j=1}^{n} \overline{G}_{ij}\right)$ % $\sum_{j=1}^{n} \overline{G}_{ij} \le t_i$
end
$\alpha = \max_{1 \le i \le n} t_i$ % $\|RA - I\|_\infty$ の上限
if $\alpha \ge 1,$ error('verification failed.'), end

式 (8) から

$$C = \mathbf{DotExact}(R_{1:k} \cdot A, 1)$$

によって得られる C は

$$|C - R \cdot A| \le \max(2^{-52} \cdot |C|, \mathit{realmin} \cdot E) =: T$$

を満たすため

$$C - I - T \le RA - I \le C - I + T$$

が成り立つ．よって，式 (7) から $\|RA - I\|_\infty$ の上限 α を計算可能となる．最後に，α が 1 より小さいかどうかを確認すれば良い．

これにより，反復の停止条件を考慮しても，1 反復あたりの高精度な行列乗算 (**DotExact**) の回数は 2 回で済むことがわかる．そして，アルゴリズム 2 によって A の正則性が保証されると，連立一次方程式 $Ax = b$ の精度保証が可能となる．

3.3 連立一次方程式の精度保証

連立一次方程式 $Ax = b$ の近似解 $\tilde{x}$ に対する残差を $r := A\tilde{x} - b$ とすると，厳密解 $x^* := A^{-1}b$ に対する $\tilde{x}$ の誤差は，連立一次方程式 $Ap = r$ の解 $p^* := A^{-1}r$ で与えられることから ($x^* =$

$\tilde{x} - p^*$)，残差反復は以下のような手順で実行できる．ただし，A が正則のとき，r の要素がすべてゼロである場合は $\tilde{x}$ が $Ax = b$ の厳密解であるため，$\|r\| \neq 0$ と仮定する．

1. 残差 $A\tilde{x} - b$ を高精度に計算する ($\tilde{r} \leftarrow A\tilde{x} - b$)．
2. 連立一次方程式 $Ap = \tilde{r}$ を解く (R を用いて $\tilde{p} \leftarrow R \cdot \tilde{r}$ とする)．
3. 近似解 $\tilde{x}$ を $\tilde{x}_{new} \leftarrow \tilde{x} - \tilde{p}$ と更新する．

我々は，この残差反復法と精度保証を組み合わせることにより，所望の相対精度 tol を与えたときに

$$\left| \frac{\tilde{x}_i - x_i^*}{\tilde{x}_i} \right| \leq \mathrm{tol}, \quad (\tilde{x}_i \neq 0 \text{ のとき}) \tag{13}$$

が成り立つような $\tilde{x} \in \mathbb{F}^n$ と

$$|\tilde{x} - x^*| \leq y$$

を満たす $y \in \mathbb{F}^n$ を求める方法を提案する．

このときに注意することは，$Ap = r$ の右辺に Δr だけ摂動を加えた連立一次方程式 $A(p + \Delta p) = r + \Delta r$ を考えると，式 (2) と同様に

$$\frac{\|\Delta p\|}{\|p^*\|} \leq \kappa(A) \frac{\|\Delta r\|}{\|r\|}$$

が成り立つことである．つまり，A の条件数を考慮して

$$\kappa(A)\|\Delta r\| \ll \|r\|$$

を満たすように，残差 $A\tilde{x} - b$ を高精度に計算し，その結果を高精度のまま保持する必要がある．

以上の議論を考慮して，Yamamoto の定理に基づく連立一次方程式の数値解の精度保証アルゴリズムを以下に示す．

アルゴリズム 3 $A \in \mathbb{F}^{n \times n}$ の正則性を判定し，正則であることが保証できたときに，式 (13) を満たす $Ax = b$ の近似解 $\tilde{x}$ と $|\tilde{x} - A^{-1}b|$ の上限 $y \in \mathbb{F}^n$ を求めるアルゴリズム：

```
function [x̃, y] = AccVerLin(A, b, tol, ℓmax)
[R_{1:k}, k, α, t] = AccInv3(A, ℓmax)          % ‖RA − I‖∞ ≤ α
x̃ = DotExact(R · b, 1)                         % x̃ ← R · b (高精度，結果は倍精度)
for loop = 1 : 10                              % 残差反復
    [r^{(1:k)}, r_rad] = DotExact(Ax̃ − b, {'midrad', k})   % |r^{(1:k)} − (Ax̃ − b)| ≤ r_rad (高精度)
    [p, p_rad] = DotExact(R_{1:k} · r^{(1:k)}, {'midrad', 1})   % |p − R_{1:k} · r^{(1:k)}| ≤ p_rad (高精度)
    q = fl△(|p| + (p_rad + (Σ_{i=1}^{k} |R_i|) · r_rad))
    y = fl△(q + (‖q‖∞/ − (α − 1)) · t)         % Yamamoto の定理，式 (11)
    relerr = fl△(max_{1≤i≤n, x_i≠0} |y_i/x_i|)   % 相対誤差の最大値
    if relerr ≤ tol                            % 残差反復の停止条件
        break
    else
        x̃ = x̃ − p                              % 近似解 x̃ の更新
    end
end
```

ここで，我々の提案するアルゴリズムの特徴を以下にまとめる．

- 事前に R の計算に必要な精度がわからなくても，反復計算により問題に応じて精度を増やしながら必要なだけの計算をする
- 高精度演算が必要なのは基本的に行列積と行列・ベクトル積のみ
- したがって高速に実装可能

次節では，数値実験によって提案方式の有効性を検証する．

4 数値実験

本節では条件数の大きい問題に対して提案方式を適用し，その性能を評価する．数値実験に使用した計算機の CPU は Pentium 4 2.53GHz である．計算はすべて MATLAB (7.0.1.24704 (R14) Service Pack 1) 上で実行した．

4.1 ヒルベルト行列

まず，条件数が 10^{16} 以上になる問題の例として，ヒルベルト行列を考える．$n \times n$ ヒルベルト行列 $H = (h_{ij})$ は，その第 (i, j) 成分を h_{ij} とするとき

$$h_{ij} = \frac{1}{i + j - 1}, \quad (1 \le i, j \le n)$$

で定義される行列である．ここでは，係数行列に誤差が含まれないように，ヒルベルト行列 H に 1 から $2n - 1$ までの公倍数 s をかけて行列 $A := s \cdot H$ を作る[4]．

右辺ベクトルについては，$z \in \mathbb{F}^n$ を $z_i = (-1)^i$ に対して $b := fl_\square(A \cdot z)$ と決める．このとき，b の計算に誤差は生じないため (すなわち，$fl_\square(A \cdot z) = Az$)，$Ax = b$ の厳密解は $x^* = z$ となる．このとき，A の条件数は $\kappa(A) = 2.45\ldots \times 10^{28}$ となる．この問題に対して，本論文で提案する精度保証法を用いて解いた例を以下に示す．ただし，アルゴリズム 3 において $\mathrm{tol} = 10^{-9}$，$\ell_{\max} = 20$ とした．

```
>> n=20; [A,b,cnd]=myhilb(n); cnd
cnd =
    2.452156585815303e+028
>> [x,y]=AccVerLin(A,b,1e-9,20); [x,y]
*** calculation of approximate inverse ***
k =
     2
*** verification for a linear system ***
loop =   1, max. rel. error = 1.095537e-004
loop =   2, max. rel. error = 8.874301e-010
ans =
   -1.000000000000000e+000    8.617077495942631e-025
    1.000000000000000e+000    9.640416868093147e-024
   -1.000000000000000e+000    1.062220917857571e-021
    1.000000000000000e+000    4.941914472774893e-020
   -9.999999999999992e-001    7.780126726211439e-016
    9.999999999999695e-001    3.054625398880036e-014
```

[4] n が大きくなると，この方法でも A は誤差を持つが，少なくとも $n \le 20$ のときは A に誤差は含まれない．

```
   -9.999999999994678e-001    5.324606289090067e-013
    9.999999999996297e-001    3.709046559563383e-013
   -9.999999999960786e-001    3.925436285241176e-012
    9.999999999602059e-001    3.980824821272961e-011
   -1.000000000120782e+000    1.208330840697098e-010
    1.000000000595174e+000    5.953338716910959e-010
   -9.999999998351981e-001    1.649529844243300e-010
    9.999999994154054e-001    5.849743981363064e-010
   -1.000000000637741e+000    6.379808828181332e-010
    9.999999991127588e-001    8.874300871906823e-010
   -9.999999999849613e-001    1.513451257318180e-011
    9.999999999542837e-001    4.574802156996649e-011
   -9.999999999890732e-001    1.093710468307201e-011
    9.999999999997991e-001    2.018749118516648e-013
```

以上の結果から，近似解 $\tilde{x}$ は残差反復により所望の精度を持つまで改善され，さらに精度保証付きで計算されていることがわかる．

次に，右辺ベクトルを $b := (1, 1, \ldots, 1)^T \in \mathbb{F}^n$ と変更した場合の結果を以下に示す．ここでは，$\mathrm{tol} = 10^{-12}$，$\ell_{\max} = 20$ とした．

```
>> b=ones(n,1);
>> [x,y]=AccVerLin(A,b,1e-12,20); [x,y]
*** calculation of approximate inverse ***
k =
     2
*** verification for a linear system ***
loop =   1, max. rel. error = 9.044724e-004
loop =   2, max. rel. error = 3.366434e-009
loop =   3, max. rel. error = 1.366729e-014
ans =
   -3.743263442685729e-015    5.116025474522475e-029
    1.493562113631585e-012    2.810456890124704e-028
   -1.478626492495270e-010    9.042732864902390e-027
    6.423810650729449e-009    8.237090906738374e-025
   -1.541714556175068e-007    2.638484718503020e-023
    2.312571834262602e-006    1.840219489496836e-022
   -2.338267076865519e-005    2.660917694689973e-021
    1.674962742815913e-004    2.209978255624679e-020
   -8.793554399783543e-004    5.504061868858120e-020
    3.463140559914753e-003    4.408457070587217e-020
   -1.038942167974426e-002    1.322862635541068e-019
    2.395577395577395e-002    3.086455593798719e-018
   -4.258804258804259e-002    1.631599120996661e-018
    5.821205821205821e-002    3.709033103548610e-018
   -6.058806058806059e-002    2.585259339885572e-018
    4.712404712404712e-002    2.010745751973094e-018
   -2.650727650727651e-002    1.998396005246811e-018
    1.018099547511312e-002    2.591511913013159e-019
   -2.388134741075917e-003    2.444648412569506e-019
```

```
    2.579979360165119e-004     2.070310897019167e-020
```

以上の結果から，近似解 $\tilde{x}$ の要素の大きさにばらつきがあったとしても，過大評価の少ない精度保証が要素毎にできていることがわかる．

4.2 より条件数の大きい行列

次に，条件数がさらに大きい問題の例として，文献 [7] にある任意に条件数を大きくできる手法を用いて得られた行列を係数行列 $A \in \mathbb{F}^{n\times n}$ とする．右辺ベクトルについては，$b := (1, 1, \ldots, 1)^T \in \mathbb{F}^n$ と決める．

$n = 100$，$\kappa(A) \approx 10^{100}$ としたときに，提案する精度保証法を用いたときの結果を以下に示す．ただし，$\mathrm{tol} = 10^{-12}$，$\ell_{\max} = 20$ とした．ここで，式 (12) を考えると，$100/16 = 6.25$ から，反復回数 k は 7〜8 回くらいになることが予測される．

```
>> n=100; A=randmat(n,1e+100); b=ones(n,1);
>> tic; [x,y]=AccVerLin(A,b,1e-12,20); toc
*** calculation of approximate inverse ***
k =
     2
k =
     3
k =
     4
k =
     5
k =
     6
k =
     7
k =
     8
*** verification for a linear system ***
loop =   1, max. rel. error = 6.921332e-006
loop =   2, max. rel. error = 4.789042e-011
loop =   3, max. rel. error = 4.264335e-016
経過時間は 8.109000 秒です
```

最後に，問題サイズをもう少し大きくして，$n = 500$，$\kappa(A) \approx 10^{50}$ としたときに，提案する精度保証法を用いたときの結果を以下に示す．ただし，$\mathrm{tol} = 10^{-12}$，$\ell_{\max} = 20$ とした．ここで，式 (12) を考えると，$50/16 = 3.125$ であるから，反復回数 k は 3〜5 回くらいになることが予測される．

```
>> n=500; A=randmat(n,1e+50); b=ones(n,1);
>> tic; [x,y]=AccVerLin(A,b,1e-12,20); toc
*** calculation of approximate inverse ***
k =
     2
k =
     3
```

```
k =
     4
k =
     5
*** verification for a linear system ***
loop =   1, max. rel. error = 3.776758e-009
loop =   2, max. rel. error = 1.023496e-016
経過時間は 190.500000 秒です
```

以上の結果から，A の条件数が非常に大きくなった場合でも，A の正則性の検証と残差反復による近似解 $\tilde{x}$ の計算及びその精度保証が可能であることが確認できた．

5 結論

本論文では，係数行列の条件数が非常に大きい場合の連立一次方程式に対する精度保証付き数値計算法を提案した．本提案方式は，基本的に内積計算の高精度演算が可能であれば IEEE754 規格に従う計算機上で実装可能である．数値実験によって，条件数が非常に大きい場合でも，所望の精度の近似解が精度保証付きで計算されることを示した．

謝辞 本研究は文部科学省 21 世紀 COE プログラム (Productive ICT Academia Program) 及び科学技術振興機構 (JST) 戦略的創造研究推進事業「シミュレーション技術の革新と実用化基盤の構築」の補助を受けた．

References

[1] U. W. Kulisch, W. L. Miranker: The arithmetic of the digital computer: A new approach, SIAM Review **28** (1986), 1–40.

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

[3] X. Li, J. Demmel, D. Bailey, G. Henry, Y. Hida, J. Iskandar, W. Kahan, S. Kang, A. Kapur, M. Martin, B. Thompson, T. Tung, and D. Yoo: Design, implementation and testing of extended and mixed precision BLAS, ACM Trans. Math. Softw., **28** (2002), 152–205.

[4] T. Ogita, S. M. Rump, S. Oishi: Accurate sum and dot product, SIAM J. Sci. Comput., to appear.

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

[6] D. M. Priest: On Properties of Floating Point Arithmetics: Numerical Stability and the Cost of Accurate Computations, PhD thesis, University of California at Berkeley, USA, 1992.

[7] S. M. Rump: A class of arbitrarily ill-conditioned floating-point matrices, SIAM J. Matrix Anal. Appl., 12:4 (1991), 645–653.

[8] S. M. Rump: Approximate inverses of almost singular matrices still contain useful information, Forschungsschwerpunktes Informations- und Kommunikationstechnik, Technical Report 90.1, Technical University Hamburg-Harburg, 1990.

[9] T. Yamamoto: Error bounds for approximate solutions of systems of equations, Japan J. Appl. Math., **1**:1 (1984), 157–171.

太田貴久 (非会員) 〒 169-0072 東京都新宿区大久保 3-14-9 シルマンホール 802

2002 年 早稲田大学大学院理工学研究科修士課程修了．修士 (理学)．2003 年 早稲田大学大学院理工学研究科博士後期課程入学，現在在学中．

荻田武史 (正会員) 〒 169-0072 東京都新宿区大久保 3-14-9 シルマンホール 802

2003 年 早稲田大学大学院理工学研究科博士後期課程了．博士 (情報科学)．2003 年 同大大学院理工学研究科客員講師．2005 年 科学技術振興機構 (JST) 研究員．小野梓記念学術賞受賞．数値計算の研究に従事．

Siegfried M. Rump (非会員) Schwarzenbergstraße 95, 21071 Hamburg, Germany

He received his Ph.D. and Habilitation from the university at Karlsruhe, Germany. He worked almost five years for the IBM development and research laboratory in Böblingen. Currently, he is Professor for Computer Science III at the Technical University at Hamburg-Harburg. His current interest is in self-validating methods, numerical analysis and matrix analysis, but also in programming languages. He wrote INTLAB, a public domain Matlab toolbox for verified computations with users in over 40 countries.

大石進一 (正会員) 〒 169-0072 東京都新宿区大久保 3-14-9 シルマンホール 802

1981 年 早稲田大学大学院理工学研究科博士後期課程了．工学博士 (早稲田大学)．1989 年 同大学教授．電子情報通信学会論文賞 (3 回)，同学会猪瀬賞，大川出版賞，丹羽記念賞, 小野梓賞，各受賞．