論　文

任意精度演算を用いた反復改良による数値計算手法

柳澤 優香

1 はじめに

線形計算とは，行列に関する数値計算のことであり，主要な問題として連立一次方程式と固有値問題の二つがある[25]．線形計算は科学技術計算の基礎を担う有用な技術であり，近年の計算機の演算能力やメモリが増大するにつれ，ますます大規模な問題を高速に解く方向へ発展している．一方，四則演算などの基本演算を計算機上で実行すると，それぞれの演算は丸め誤差を伴った浮動小数点演算によって近似されるため，計算の際にどのくらいの丸め誤差が発生し累積し伝播するかを考慮する必要がある．例えば，密行列で 100 万元を超えるような大規模な行列を扱う場合，理論的には倍精度では計算精度が不足し，十分な有効桁数が得られなくなる可能性がある．また，特異に近い行列の場合，ごく小さな誤差による摂動に対して解の相対変化が大きくなり不安定である可能性が高い．何らかの方法で得られた数値計算の結果が一体どの程度の許容誤差の範囲内で正しいかということを示すことは計算自体と同様に重要である．例えば，演算桁数を多くして再度試すことで，丸め誤差の見当を大雑把につけることはできるかもしれないが，計算時間とメモリの両面でコストが大きい．それが大規模な問題に対してであれば莫大な計算時間がかかることが予想できる．そこで近年，誤差解析に基づき必要な部分のみを任意精度演算で行う計算手法に関する研究が活発である(例えば，[1])．本稿で紹介する数値計算手法は，

- 任意の精度を達成する高精度な内積計算手法
- 標準的な数値計算ライブラリ (LAPACK や BLAS など)

で構成され，精度保証付き数値計算の適用可能性を拡げる重要な技術である．

「精度保証付き数値計算」とは，得られた近似解の周りに真の解が存在すること，及びその誤差範囲を保証する数値計算手法をいう[7, 11]．単純な方法として，実数の四則演算を区間に対する四則演算(区間演算)に置き換えると，計算された結果は真の解を含み，区間幅が誤差となる．しかし，この方法論の単純な適用は有用ではなく，「通常の数値計算によって近似解を得た後に，区間演算によってその近似解の精度を保証する」[11]という方針に基づき，計算複雑度(計算量)，計算精度

[筆者紹介]

やなぎさわ ゆうか．早稲田大学理工学術院総合研究所研究院講師．2013 年 3 月，東京女子大学大学院理学研究科博士前期課程を修了．2016 年 3 月，早稲田大学大学院基幹理工学研究科博士後期課程を修了し，博士(工学)を取得．2016 年 4 月より早稲田大学理工学術院総合研究所に所属し，2017 年 6 月より現職．大学院在学時に荻田武史教授(東京女子大学)らが従事する高精度な数値計算手法に関する研究に興味を持つ．現在，大石進一教授(早稲田大学)らと共に主に線形問題に対する精度保証付き数値計算法について研究しており，より広い分野へ適用可能性を拡げていければと考えている．日本応用数理学会会員，SIAM 会員．

などの観点で改良が重ねられてきた．詳細は[11]を参照されるとよい．本稿では，通常の倍精度演算の範囲で扱うことが難しい悪条件問題にフォーカスし，高精度な結果が得られる Rump の方法[15]を解説する．Cholesky 分解の逆行列を使った手法[9]と，近年筆者らが取り組んだ収束解析[26,27]について説明する．また，与えられた悪条件な実対称行列の正定値性を検証する時，多倍長精度演算[5,6]を使った場合よりも数倍高速であることを数値例により示す．

2 近似逆行列を用いた反復改良

$A \in \mathbb{R}^{n\times n}$ は正則とする．

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

を A の条件数といい，問題の難しさを表す指標で $\kappa(A)$ が大きいと悪条件(ill-conditioned)な問題という．ただし，ノルムを 2 ノルム，無限大ノルムなど一つ定め，$\|\cdot\|$ で表すことにする．例えば，A の近似逆行列の計算において，A が誤差により $A+\delta A$ に微小に変化した場合を考える．このとき得られる結果を $A^{-1}+\Delta A$ とすると，$\|A^{-1}\delta A\|<1$ ならば，

$$\frac{\|\Delta A\|}{\|A^{-1}\|} \le \frac{\|A^{-1}\|\cdot\|\delta A\|}{1-\|A^{-1}\delta A\|} \qquad (1)$$

となる．(1)の分子について，$0<\varepsilon\ll 1$，$\|\delta A\|:=\varepsilon\|A\|$ とおくと，A^{-1} の相対誤差は微小な ε のおよそ $\kappa(A)$ 倍であることを示している[2]．IEEE754 規格の倍精度浮動小数点演算の場合，仮数部の相対精度は $2^{-53}=1.1\times10^{-16}$(これを $\mathbf{u}$ と定義する)[4]なので，

$$\kappa(A) > \mathbf{u}^{-1} \qquad (2)$$

を満たす場合，(1)より理論的には正しい計算結果が得られないことがわかる．具体的に，$A \in \mathbb{R}^{n\times n}$，$b \in \mathbb{R}^n$ の連立一次方程式 $Ax=b$ において，何らかの方法で近似解 $\tilde{x}$ が得られたとする．その近似解の近くに真の解が存在するのか，得られた $\tilde{x}$ は精度良く求められているのかを調べたい時，最も基本的な方法としては，A の近似逆行列 $C \in \mathbb{R}^{n\times n}$ が存在するとしたら，

$$\|I-CA\| < 1 \qquad (3)$$

を満たすかを確認する．ただし，I は $n\times n$ の単位行列を表す．(3)を厳密に満たせば行列 A は正則であり，得られた近似解に対する厳密な誤差範囲が得られるが，もし A が(2)の場合，上で考察した通り C が真の逆行列の情報はほとんど持たないため，たとえ A が正則行列であっても(3)で数値的に破綻してしまい保証ができない可能性が高い．行列 A が正則であることを保証可能にし，解のシャープな存在範囲を特定するためには，精度が良い近似逆行列が必要である．

1990 年，Rump は悪条件行列の近似逆行列は，前処理行列として有益な情報を持っていることを発見し，その性質を応用した手法を開発した[15](以降，Rump の方法と呼ぶ)．例えば，以下の 4×4 行列 A[16]を考える．

$$A = \begin{pmatrix} 177830 & 3777 & 112815 & 6116 \\ 3777 & 28534 & 32741 & 1890 \\ 112815 & 32741 & 128870 & 7095 \\ 6116 & 1890 & 7095 & 391 \end{pmatrix}$$

$\kappa(A)=1.1\cdot10^{19}$ であり，A の逆行列 A^{-1} は

$$\begin{pmatrix} 8.454\cdot10^{6} & 1.516\cdot10^{8} & 1.752\cdot10^{9} & -3.266\cdot10^{10} \\ 1.516\cdot10^{8} & 2.720\cdot10^{9} & 3.143\cdot10^{10} & -5.859\cdot10^{11} \\ 1.752\cdot10^{9} & 3.143\cdot10^{10} & 3.631\cdot10^{11} & -6.769\cdot10^{12} \\ -3.266\cdot10^{10} & -5.859\cdot10^{11} & -6.769\cdot10^{12} & 1.261\cdot10^{14} \end{pmatrix}$$

となるが，浮動小数点数を用いて A の逆行列を計算†1して得られた結果は，

$$\begin{pmatrix} -1.285\cdot10^{7} & -2.306\cdot10^{8} & -2.664\cdot10^{9} & 4.967\cdot10^{10} \\ -2.306\cdot10^{8} & -4.137\cdot10^{9} & -4.780\cdot10^{10} & 8.910\cdot10^{11} \\ -2.664\cdot10^{9} & -4.780\cdot10^{10} & -5.523\cdot10^{11} & 1.029\cdot10^{13} \\ 4.967\cdot10^{10} & 8.910\cdot10^{11} & 1.029\cdot10^{13} & -1.9189\cdot10^{14} \end{pmatrix}$$

となる．真の逆行列と比較すると最大で 10 倍のオーダーで値が異なり，(3)は満たさない．Rump[15]は，このような $C^{(1)}:=\mathrm{inv}(A)$ を最初の前処理行列とし，反復改良することで，**表 1** の通り(3)を満たす $C^{(k)}$ を得ることができると述べている．

†1 LAPACK では xGETRF で LU 分解した後，xGETRI で逆行列を求める．Matlab のコマンドは inv(A)．

表 1 テスト行列 A の実験結果：$\kappa(A)\approx 1.1\cdot 10^{19}$

k	$\kappa(C^{(k)}A)$	$\|I-C^{(k)}A\|_\infty$
1	$2.632\cdot 10^{4}$	$1.823\cdot 10^{2}$
2	1.00	$8.716\cdot 10^{-15}$

表 1 の通り，$\kappa(C^{(1)}A)$ は $n\mathbf{u}\cdot\kappa(A)$ ほど条件数が改善されており，前処理行列として機能していることがわかる．Rump の方法の特徴は反復ごとに行列積の演算精度を増やしながら (3) を満たすまで反復改良を行う手法であるため，問題に適した計算精度を自分で予想する必要がない．任意精度を達成する行列積の計算が高速になったことから，Rump の方法を基にした，連立一次方程式に対する精度保証付き数値計算法[13]が確立した．A の条件数が非常に大きい場合でも，(3) を満たす $C^{(k)}$ とそれを使った近似解の逐次補正により高精度な近似解 $\tilde{x}$ とその誤差範囲が得られる．また，Rump の方法の収束性は厳密には証明が困難であったが，問題となる行列にある一定量の摂動を加えることで丸め誤差の影響で問題の条件数が改善し，数回の反復で収束する仕組みを [12] で説明しており，さらに [18] では，収束するのに必要な演算精度を誤差解析により導出している．

3 逆 Cholesky 分解を用いた反復改良

LU 分解，Cholesky 分解，QR 分解，特異値分解など与えられた行列を考察しやすい行列の積として表すことは線形計算の基本であり，連立一次方程式や行列の固有値問題を取り扱う場合の基礎となる．2 章の Rump の方法を基にした LU 分解と QR 分解の逆行列を高精度に求める手法[8]が 2010 年に提案された．例えば QR 分解の場合，問題の悪条件性は上三角行列に反映されるため，その近似逆行列は前処理行列として有益であることが示されている．同様に Rump の方法を基にした (筆者らによる) Cholesky 分解の逆行列 (上三角行列) を高精度に求める手法[9, 26, 27]を本章では解説する．Cholesky 分解は連立一次方程式，一般化固有値問題の解法のみならず，実対称行列に対する正定値性の判定など幅広く用いられる手法であり，精度の良い逆行列を得ることが必要である．実対称行列 $B=(b_{ij})\in\mathbb{R}^{n\times n}$ が正定値であれば $B=R^TR$ ($R\in\mathbb{R}^{n\times n}$ は上三角行列) の形に分解できる．具体的な算法は次のように与えられる．

Algorithm 1 Cholesky 分解

for $j=1{:}n$

for $i=1{:}j-1$

$$r_{ij}=\left(b_{ij}-\sum_{k=1}^{i-1}r_{ki}r_{kj}\right)/r_{ii}$$

end

$$r_{jj}=\left(b_{jj}-\sum_{k=1}^{j-1}r_{kj}^2\right)^{1/2} \tag{4}$$

end

実対称行列 $B\in\mathbb{R}^{n\times n}$ が任意の $x\in\mathbb{R}^n$ に対し $x^TBx>0$ となるとき，B は正定値であるといい，B の Cholesky 分解が存在することと同値である．しかし実際は，B が正定値であっても丸め誤差により (4) で負の数の平方根が現れ，R が得られない (これを数値的に破綻すると呼ぶ)．悪条件な行列の場合，演算桁数を増やして計算する必要がある．次に紹介する手法は一般的な数値計算ライブラリ†2と任意精度を達成する内積計算を使って Cholesky 分解の逆行列を高精度に計算する手法である．具体的には，R の近似逆行列 X を $X=\sum_{i=1}^{k}X_i$, $X_i\in\mathbb{F}^{n\times n}$ のように行列 X_1 から X_k の和で表現し，$\|X^TBX-I\|<\varepsilon_{tol}\le 1$ を満たすまで反復改良を行う方法である．

3.1 シフト付き Cholesky 分解

$B\in\mathbb{F}^{n\times n}$ を実対称正定値行列とし†3,

$$\kappa(B)>\mathbf{u}^{-1}, \tag{5}$$

$$(n+1)(n+3)\mathbf{u}<1. \tag{6}$$

と仮定する．ただし，$\mathbb{F}$ は浮動小数点数の集合

†2 LAPACK では xPOTRF. Matlab のコマンドは chol(A).

†3 入力誤差が大きいと元々の行列の正定値性が失われてしまうことがあるため，行列 B を要素が浮動小数点数である行列の和で表現することも考慮する．例として 3.4 節の数値実験で使っているテスト行列がある．

を意味し，$\mathbb{F}\subset\mathbb{R}$ である．$fl(\cdot)$ は括弧内の演算を浮動小数点演算で評価することを意味し，$a, b\in\mathbb{F}$ について，$\circ\in\{+, -, *, /\}$ とすると，

$$fl(a\circ b) = (1+\varepsilon)(a\circ b), \quad |\varepsilon_1|\le\mathbf{u} \quad (7)$$

となる[10]．また，$e, f\in\mathbb{R}$ について $|e|=\mathcal{O}(1)\cdot|f|$ を $\approx$ と表記する．(5)の場合，Cholesky 分解が数値的に破綻し計算できない場合があるため，B の対角成分に正方向に摂動 δ を加えることを考える．$\lambda_{\min}(B)$ と $\lambda_{\max}(B)$ をそれぞれ B の最小，最大固有値とすると，$\delta\le\lambda_{\min}(B)$ のように δ が相対的に微小の場合は，

$$\kappa(B+\delta I) = \frac{\lambda_{\max}(B)+\delta}{\lambda_{\min}(B)+\delta} \approx \frac{\lambda_{\max}(B)}{\lambda_{\min}(B)} = \kappa(B)$$

に丸められる．一方，$\lambda_{\min}(B)<\delta$ の場合，

$$\kappa(B+\delta I) = \frac{\lambda_{\max}(B)+\delta}{\lambda_{\min}(B)+\delta} \approx \frac{\lambda_{\max}(B)}{\delta} \quad (8)$$

のように分母は δ に丸められる．$\alpha:=\delta/\|B\|_2$ とおくと(8)の値は α^{-1} と表せる．つまり，浮動小数点演算による丸め誤差の影響で摂動 δ は悪条件行列 B の条件数を減らす効果が期待できる．つまり，

$$\kappa(B+\delta I) \approx \min(\kappa(B), \alpha^{-1})$$

となる．次に，δ を Cholesky 分解が破綻しないための適切な大きさに設定する必要がある(計算誤差を最小限に抑えるために可能な限り小さいことが望ましい)．Cholesky 分解の後退誤差の上限は[3, 17, 20]で次の通り示されている．トレース $\sum_{i=1}^{n} b_{ii}$ を $\mathrm{Tr}(B)$ と表記し，B の Cholesky 分解を実行し $\hat{R}$ を得た場合，$\hat{R}$ は

$$\hat{R}^T\hat{R} = B+\Delta B, \quad \|\Delta B\|_2 \le c_n'\mathbf{u}\mathrm{Tr}(B)$$

を満たす．ただし，$\Delta B\in\mathbb{F}^{n\times n}$, $c_n':=\dfrac{(n+1)}{1-(n+1)\mathbf{u}}$ とし，仮定(6)より $(n+1)\mathbf{u}<1$ を満たしている．また，B の最小固有値と Cholesky 分解の関係性について次の定理がある[21]．

補助定理 1(定理 2. 3[21]**)** $B\in\mathbb{F}^{n\times n}$ は実対称行列で，$\lambda_{\min}(B)>c_n'\mathbf{u}\mathrm{Tr}(B)$ であれば，Cholesky 分解は成功する．

補助定理 1 を基に，適切なシフト量が[27]において次のように議論されている．$B=(b_{ij})\in\mathbb{F}^{n\times n}$ は実対称で正定値と仮定する．仮定より B の対角成分 b_{ii} は，$b_{ii}>0$ となる．b_{ii} にある $\delta(0<\delta<1)$ を加えたものを $\widetilde{B}:=fl(B+\delta I)$ とし丸め誤差を考慮すると，(7)より次のように表せる．

$$\widetilde{B} = B+\delta I+D, \quad \|d_{ii}\| \le \mathbf{u}(b_{ii}+\delta). \quad (9)$$

ただし，$D=(d_{ij})\in\mathbb{F}^{n\times n}$ とする．補助定理 1 より $\lambda_{\min}(\widetilde{B})\ge c_n'\mathbf{u}\cdot\mathrm{Tr}(\widetilde{B})$ であれば Choleksy 分解は数値的に破綻しないため，まず，$\lambda_{\min}(\widetilde{B})$ の下限を求める．Weyl の定理(例えば[21]の Corollary 4. 9)と(9)より $\lambda_{\min}(\widetilde{B})$ の下限は，

$$\lambda_{\min}(\widetilde{B}) \ge \lambda_{\min}(B+\delta I)-\|D\| > \delta-\max_{1\le i\le n}|d_{ii}|$$

となる．ここで，$\max_{1\le i\le n}|d_{ii}|$ を $\mathrm{Tr}(B)$ を使って表すと，

$$\begin{aligned}\max_{1\le i\le n}|d_{ii}| &\le \max_{1\le i\le n}\mathbf{u}(b_{ii}+\delta)\\ &= \mathbf{u}\max_{1\le i\le n} b_{ii}+\delta\mathbf{u} \le \mathbf{u}\cdot\mathrm{Tr}(B)+\delta\mathbf{u}\end{aligned}$$

となり，$\lambda_{\min}(\widetilde{B})$ の下限は，

$$\begin{aligned}\lambda_{\min}(\widetilde{B}) &\ge \delta-(\mathbf{u}\cdot\mathrm{Tr}(B)+\delta\mathbf{u})\\ &= (1-\mathbf{u})\delta-\mathbf{u}\cdot\mathrm{Tr}(B) \quad (10)\end{aligned}$$

と表せる．一方，$c_n'\mathbf{u}\cdot\mathrm{Tr}(\widetilde{B})$ は(9)より，

$$\begin{aligned}c_n'\mathbf{u}\cdot\mathrm{Tr}(\widetilde{B}) &= c_n'\mathbf{u}\cdot\mathrm{Tr}(B+\delta I+D)\\ &\le c_n'\mathbf{u}\cdot\mathrm{Tr}(B+\delta I+|D|)\\ &\le c_n'\mathbf{u}(1+\mathbf{u})(\mathrm{Tr}(B)+n\delta)\end{aligned}$$

と表せるので，この結果と(10)より

$$(1-\mathbf{u})\delta-\mathbf{u}\cdot\mathrm{Tr}(B) \ge c_n'\mathbf{u}(1+\mathbf{u})(\mathrm{Tr}(B)+n\delta)$$

を満たす δ が $\widetilde{B}$ の Cholesky 分解が成功するために必要なシフト量である:

$$\delta \ge \frac{(1+\mathbf{u})c_n'+1}{1-((1+\mathbf{u})nc_n'+1)\mathbf{u}}\mathbf{u}\cdot\mathrm{Tr}(B). \quad (11)$$

(11)を計算し，便宜上簡単に表すと，$\delta\ge c_n\mathbf{u}\cdot\mathrm{Tr}(B)$,

$$c_n := \frac{n+2}{1-(n+1)(n+3)\mathbf{u}}$$

となる．以上のことから[26]で次の定理が示されている．

定理 1 $B\in\mathbb{F}^{n\times n}$ を実対称行列とし，(6)と仮定する．

$$\begin{aligned}shift(B) &:= c_n\mathbf{u}\cdot\mathrm{Tr}(B),\\ c_n &= \frac{n+2}{1-(n+1)(n+3)\mathbf{u}}\end{aligned}$$

と定義し，対角シフト量を $\delta:=shift(B)$ とおく．B が正定値であれば，$\widetilde{B}:=fl(B+\delta I)$ の Cholesky 分解は成功する．

3.2 任意精度の内積計算

一般的に多倍長演算は倍精度演算に比べて当然低速であるが，内積計算については比較的高速な方法が現在までに多数開発されている．例えば計算機環境に依存しない，無誤差変換による加減乗算を用いた高速な任意精度内積演算アルゴリズムとして[10, 22, 23]などが Ogita らにより提案されている．その他にも Julia の任意精度演算のための BigFloat 型，MATLAB の Multiprecision Computing Toolbox [6]がある．本稿で扱う手法は内積計算手法に依存しないが，内積の値を必要な精度で得ることができるという仮定のもとに議論を進める．$X\in\mathbb{F}^{n\times n}$, $D_{1:l}:=\sum_{i=1}^{l}D_i$, $D_i\in\mathbb{F}^{n\times n}$ について

$$|BX-D_{1:l}| \le c_1\mathbf{u}^l|BX|+c_2\mathbf{u}^k|B||X| \quad (12)$$

を満たすような任意精度の内積計算(k 倍精度で計算し l 倍精度に丸める)が必要で

$$D_{1:l} := \{BX\}_k^l$$

と記述する．c_1, c_2 は $\mathcal{O}(1)$ の定数とする．また，$\langle G, E\rangle$ は中心 $G=G^T\in\mathbb{F}^{n\times n}$, 半径 $E\in\mathbb{F}^{n\times n}$ の中心半径型の区間行列を意味し，k 倍精度で区間演算し倍精度の区間行列に丸めることを

$$\langle G, E\rangle := \{X^TBX\}_k^1$$

と記述する．このとき $X^TBX\in\langle G, E\rangle$ を満たす．実際の計算では，(12)のように

$$|X^TBX-G| \le c_3\mathbf{u}|X^TBX|+c_4\mathbf{u}^k|X^T\|B\|X| =: E$$

を満たす任意精度演算[†4]を行う．c_3, c_4 は $\mathcal{O}(1)$ の定数とする．

3.3 反復改良による高精度な計算手法

仮定(5), (6)を満たす $B\in\mathbb{F}^{n\times n}$ に対して

$$\delta_1 := shift(B), \quad \alpha_1 := \frac{\delta_1}{\|B\|_2} \le n^2\mathbf{u},$$
$$(\widetilde{R}^{(1)})^T\widetilde{R}^{(1)} = (B+\delta_1 I)+\Delta_1,$$
$$X^{(1)} := \mathrm{inv}(\widetilde{R}^{(1)})$$

で得た近似逆行列 $X^{(1)}\in\mathbb{F}^{n\times n}$ が前処理行列として有益であることを利用すると，得られた $X^{(1)}$ は

$$\kappa(X^{(1)T}BX^{(1)}) \approx \alpha_1\cdot\kappa(B)$$

のように前処理することで条件数が改善することを[7]で示している．ただし，$X^{(1)T}BX^{(1)}$ の計算は $X^{(1)T}BX^{(1)}\in\langle G^{(1)}, E^{(1)}\rangle$ となるような区間演算 $\langle G^{(1)}, E^{(1)}\rangle:=\{X^{(1)T}BX^{(1)}\}_2^1$，つまり 4 倍精度で計算し倍精度に丸める必要があり，$\|E^{(1)}\|_2\approx\mathbf{u}$ である．Weyl の定理[24]より

$$|\lambda_i(X^{(1)T}BX^{(1)})-\lambda_i(G^{(1)})| \le \|E^{(1)}\|_2$$

がいえるので，$G^{(1)}$ の正定値性を失わないよう対角成分 $G_{ii}^{(1)}$ に対して $S_{ii}^{(1)}:=G_{ii}^{(1)}+\|E^{(1)}\|_\infty$ とした $S^{(1)}$ を使って，次のステップに進む．

$$\delta_2 := shift(S^{(1)}), \quad \alpha_2 := \frac{\delta_2}{\|S^{(1)}\|_2} \le n^2\mathbf{u},$$
$$(\widetilde{R}^{(2)})^T\widetilde{R}^{(2)} = (S^{(1)}+\delta_2 I)+\Delta_2,$$
$$T := \mathrm{inv}(\widetilde{R}^{(2)}),$$
$$X_{1:2}^{(2)} := \{X^{(1)}T\}_2^2.$$

得られた $X^{(2)}$ について

$$\kappa(X_{1:2}^{(2)T}BX_{1:2}^{(2)}) \approx \prod_{i=1}^{2}\alpha_i\cdot\kappa(B)$$

のようにさらに条件数が改善する．前処理は区間演算 $\langle G^{(2)}, E^{(2)}\rangle:=\{X_{1:2}^{(2)T}BX_{1:2}^{(2)}\}_3^1$，つまり 6 倍精度で計算し倍精度に丸める必要があり，$\|E^{(2)}\|_2\approx\mathbf{u}$ となる．同様の手順で k 反復目に得られた $X_{1:m_k}^{(k)}$, $m_k:=\lceil\frac{k}{2}\rceil+1$ が

$$\kappa\left(X_{1:m_k}^{(k)T}BX_{1:m_k}^{(k)}\right) \approx \prod_{i=1}^{k}\alpha_i\cdot\kappa(B)$$

を満たすと仮定すると，$k+1$ 反復目は，

$$S_{ii}^{(k)} := G_{ii}^{(k)}+\|E^{(k)}\|_\infty,$$
$$\delta_{k+1} := shift(S^{(k)}),$$
$$\alpha_{k+1} := \frac{\delta_{k+1}}{\|S^{(k)}\|_2} \le n^2\mathbf{u},$$
$$(\widetilde{R}^{(k+1)})^T\widetilde{R}^{(k+1)} = (S^{(k)}+\delta_{k+1}I)+\Delta_{k+1},$$
$$T := \mathrm{inv}(\widetilde{R}^{(k+1)}),$$

†4 実際は，$F_{1:k}=\sum_{i=1}^{k}F_i$, $F_i\in\mathbb{F}^{n\times n}$, $E_1\in\mathbb{F}^{n\times n}$ について，$\langle F_{1:k}, E_1\rangle:=\{BX\}_k^k$, $\langle G, E\rangle:=\{X^TF_{1:k}\}_k^1$ と計算する．

$$X_{1:m_{k+1}}^{(k+1)} := \{X^{(k)}T\}_{m_{k+1}}^{m_{k+1}}$$

で得られた $X_{1:m_{k+1}}^{(k+1)}$ について

$$\kappa(X_{1:m_{k+1}}^{(k+1)T} B X_{1:m_{k+1}}^{(k+1)}) \approx \alpha_{k+1}\cdot\kappa(X_{1:m_k}^{(k)T} B X_{1:m_k}^{(k)})$$

を満たすことを帰納法によって証明した．つまり，各ステップで必要な精度を達成する内積演算が可能であれば，ある一定の反復回数で前処理行列の条件数が 1 に収束し，精度の良い B の近似逆行列を得ることができる．また，誤差解析により，内積演算に必要な精度は最小限に近いことを[26]で示している．

次のアルゴリズムは $B\in\mathbb{F}^{n\times n}$ に対して

$$\|X_{1:m_k}^{(k)T} B X_{1:m_k}^{(k)} - I\|_2 < \varepsilon_{tol}$$

を満たす上三角行列 $X_{1:m_k}^{(k)}$ を計算する．ただし，許容誤差 ε_{tol} は $\varepsilon_{tol}\leq 1$，$k\in\mathbb{N}$ は反復回数とする．

Algorithm 2（高精度逆 Cholesky 分解）

$k=0$，$G^{(0)} := A_{1:1}$，$E^{(0)} := O$，$X_{1:1}^{(0)} := I$

repeat

$k = k+1$

$R^{(k)T}R^{(k)} := S^{(k)}$

if Cholesky factorization fails

$S_{ii}^{(k)} := G_{ii}^{(k-1)} + \|E^{(k-1)}\|_\infty$

$\delta_k := shift(S^{(k)})$

$R^{(k)T}R^{(k)} := fl(S^{(k)}+\delta_k I)$

if end

$T^{(k)} := \mathrm{inv}(R^{(k)})$

$X_{1:m_k}^{(k)} := \left\{X_{1:m_{k-1}}^{(k-1)} T^{(k)}\right\}_{mk}^{mk}$ $\quad //m_k := \lceil \frac{k}{2} \rceil + 1$

$\langle G^{(k)}, E^{(k)}\rangle := \left\{X_{1:m_k}^{(k)T} A X_{1:m_k}^{(k)}\right\}_{k+1}^{1}$

until $\|G^{(k)}-I\|_\infty + \|E^{(k)}\|_\infty < \varepsilon_{tol}$

3.4 数値実験

本節では Algorithm 2 が悪条件行列 B に効果的に作用し，収束することを検証するため，数値実験を行う．計算環境は，下記の通りである．

CPU: Intel Xeon CPU E5-2690, 2.60 GHz,
24 Core
Memory: 256 GB
OS: CentOS 6.6
Software: MATLAB R2017a

表 2 $n=21$, $\kappa(B)\approx 8.16\cdot 10^{29}$

k	$\kappa(\hat{G}^{(k)})$	$\|I-\hat{G}^{(k)}\|_\infty$	α_k
1	$2.72\cdot 10^{15}$	$1.00\cdot 10^{0}$	$3.34\cdot 10^{-15}$
2	$1.19\cdot 10^{0}$	$1.87\cdot 10^{-1}$	–
3	$1.00\cdot 10^{0}$	$3.72\cdot 10^{-16}$	–

表 3 $n=500$, $\kappa(B)\approx 9.9\times 10^{48}$

k	$\kappa(\hat{G}^{(k)})$	$\|I-\hat{G}^{(k)}\|_\infty$	α_k
1	$2.75\cdot 10^{36}$	$1.00\cdot 10^{0}$	$2.75\cdot 10^{-13}$
2	$1.97\cdot 10^{25}$	$1.00\cdot 10^{0}$	$7.16\cdot 10^{-12}$
3	$2.66\cdot 10^{14}$	$1.00\cdot 10^{0}$	$1.35\cdot 10^{-11}$
4	$1.00\cdot 10^{0}$	$3.00\cdot 10^{-3}$	–
5	$1.00\cdot 10^{0}$	$1.08\cdot 10^{-15}$	–

許容誤差 $\varepsilon_{tol} := 10^{-6}$ とし，**表 2** は Hilbert 行列†5，**表 3** は条件数 10^{cnd}，$\mathrm{cnd}:=50$ として生成したテスト行列†6に Algorithm 2 を適用した結果である．$\hat{G}^{(k)} := X_{1:m_k}^{(k)T} B X_{1:m_k}^{(k)}$ とする．

表 2, 3 の通り，$\kappa(\hat{G}^{(k)}) > \mathbf{u}^{-1}$ の間の収束速度は $\prod_{i=1}^{k}\alpha_i\cdot\kappa(B)$ で，$\kappa(\hat{G}^{(k)})\leq\mathbf{u}^{-1}$ まで改善されるとシフトなしで Cholesky 分解を行い収束することが観察できる．

3.5 実対称行列に対する正定値性の検証

与えられた実対称行列 $B\in\mathbb{F}^{n\times n}$ が厳密に正定値であるかを Algorithm 2 で判定するまでの計算時間を測定する．$\varepsilon_{tol}:=1$ に設定する．一方で，浮動小数点演算による Cholesky 分解が成功するかどうかだけで正定値性が判定できる Rump の方法[17, 20]がある．

定理 2（Corollary 2.4[17]**）** 実対称行列 $B\in\mathbb{F}^{n\times n}$ に対して，$\widetilde{B}\leq B-cI$，$c:=\sum_{j=1}^{n}\gamma_{j+1}b_{jj}$，$\gamma_j:=$

†5 入力誤差のため Hilbert 行列の条件数は高々 $\mathbf{u}^{-1}$ であるが，$B=(b_{ij})\in\mathbb{F}^{n\times n}$ について

$$b_{ij} := lcm(1, 2, \cdots, 2n-1)/(i+j-1)$$

のように各要素を分母の最小公倍数でスケーリングして得た行列を使用する．

†6 条件数 cnd と指定し，ランダムな直交行列 $Q\in\mathbb{F}^{n\times n}$，$10^{1}$ と 10^{cnd} の間で対数的に等間隔な n 点のベクトルを対角成分に持つ対角行列 $D\in\mathbb{F}^{n\times n}$ からテスト行列 B を $B_{1:l} := QDQ^T$ と生成する．ただし，$l=\lceil \mathrm{cnd}/\log_{10}\mathbf{u}^{-1}\rceil$ とする．

表 4　Algorithm 2 と MPFR を用いた場合とを比較

	Algorithm 2		定理 2(MPFR)	
n	t_1	k	t_2	$\frac{d}{53}$
1000	1.373	2	1.919	2
3000	18.438	2	21.580	2
5000	61.158	2	82.330	2
7000	132.753	2	210.269	2
10000	303.060	2	584.929	2

$j\mathbf{u}/(1-j\mathbf{u})$ が与えられ，$\widetilde{B}$ の Cholesky 分解が成功すれば，B は正定値である．

表 4 は，定理 2 を MPFR[5, 6]を介して計算し，判定が成功するまでの計算時間を測定し，Algorithm 2 と比較したものである．MPFR の計算精度 2^{-d} は $d=53$ からスタートし，失敗したら計算精度を 2 倍にする．t_i は実行時間を示し単位は秒である．また，テスト行列は[16]を使用し条件数は 10^{19} に固定した．

表 4 の通り，$n\geq 1000$ の場合 t_1 は t_2 より小さく特に $n\geq 7000$ の場合 2 倍近く高速であることが観察できる．t_2 は演算量 $\mathcal{O}(n^3)$ に依存して増えているが，一方 t_1 の場合，行列サイズが大きくなるにつれて，行列積†7の演算で最適化の効果が得られ性能が向上していることがわかる．

4 むすび

本稿において，解ける問題は計算精度によって必ずしも制限されるものではないことを示した．必要な箇所のみ任意精度演算を行い，反復改良を活用することで，倍精度演算の範囲で扱うことが難しい悪条件問題(条件数が $\mathbf{u}^{-1}$ より大きい問題)に対して精度保証を行うことが可能である．さらに，3.5 節において，Algorithm 2 はすべての計算を多倍長精度演算で行う場合よりも高速であることを示した．

近年，気候や気象のモデリングや機械学習において，倍精度の代わりに単精度や半精度などのより低い精度を使用することへの関心が高まっている[1]．計算精度を低くすれば，計算環境によってはパフォーマンスの大幅な向上やエネルギー消費を抑えることが期待できるが，解くべき問題が計算精度に対して悪条件になる場合に注意する必要がある．本稿で解説した方法は，他の $\mathbf{u}$(例えば，$2^{-24}\approx 6.0\cdot 10^{-8}$)にも当てはめることができ，必要な箇所のみ計算精度を上げれば，計算精度に対して悪条件な問題でも扱うことが可能である．本稿では基礎的な手法を解説したが，特に前述した観点において，将来の発展や異分野への応用につながる研究と考えている．

†7 Matlab で採用されている BLAS や LAPACK のルーチンは，Intel Math Kernel Library によって提供されている．

謝 辞

本稿の執筆にあたり，早稲田大学理工学術院大石進一教授より貴重な助言を頂いた．感謝の意を表したい．また，本研究は早稲田大学理工学術院総合研究所の助成を受けたものである．

参考文献

[1] Carson, E. and Higham, N. J., Accelerating the solution of linear systems by iterative refinement in three precisions, SIAM Journal on Scientific Computing, 40[2] (2018), A817–A847.
[2] Demmel, J. W., on condition numbers and the distance to the nearest ill-posed problem, Numer. Math. 51 (1987), 251–289.
[3] Demmel, J. W., On floating point errors in Cholesky, LAPACK Working Note 14 CS–89–87, Department of Computer Science, University of Tennessee, Knoxville, TN, USA, 1989.
[4] Higham, N. J., Accuracy and Stability of Numerical Algorithms, 2nd ed., SIAM, Philadelphia, PA, 2002.
[5] MPFR, The multiprecision floating-point reliable library, http://www.mpfr.org/
[6] Multiprecision Computing Toolbox for MATLAB, https://www.advanpix.com/
[7] 中尾充宏，渡部善隆，実例で学ぶ精度保証付き数値計算 理論と実装，サイエンス社，2011.
[8] Ogita, T., Accurate matrix factorization: inverse LU and inverse QR factorizations, SIAM J. Matrix Anal. Appl., 31(2010), 2477–2497.
[9] Ogita, T. and Oishi, S., Accurate and robust inverse Cholesky factorization, Nonlinear Theory and Its Applications, IEICE 3(2012), 103–111.
[10] Ogita, T., Rump, S. M. and Oishi. S., Accurate sum and dot product, SIAM J. Sci. Comput. 26 (2005),

1955–1988.
[11] 大石進一，精度保証付き数値計算の基礎，コロナ社，2018.
[12] Oishi, S., Tanabe, K., Ogita, T. and Rump, S. M., Convergence of Rump's method for inverting arbitrarily ill-conditioned matrices, J. Comp. Appl. Math. 205[1] (2007), 533–544.
[13] 太田貴久，荻田武史，Rump, S. M., 大石進一，悪条件連立一次方程式の精度保証付き数値計算法，日本応用数理学会論文誌，15[3](2004), 269–286.
[14] Ozaki, K., Ogita, T., Oishi, S. and Rump, S. M., Error-free transformations of matrix multiplication by using fast routines of matrix multiplication and its applications, Numer. Algorithms 59(2012), 95–118.
[15] Rump, S. M., Approximate inverses of almost singular matrices still contain useful information, Forschungsschwerpunkt Informations und Kommunikationstechnik, TUHH, Technical Report 90.1(1990).
[16] Rump, S. M., A class of arbitrarily ill-conditioned floating-point matrices, SIAM J. Matrix Anal. Appl. 12 [4](1991), 645–653.
[17] Rump, S. M., Verification of positive definiteness, BIT Numerical Mathematics, 46, 433–452(2006).
[18] Rump, S. M., Inversion of extremely ill-conditioned matrices in floating-point, Japan J. Indust. Appl. Math., 26(2009), 249–277.
[19] Rump, S. M., INTLAB–INTerval LABoratory, Version 10.2, 2017.
[20] Rump, S. M. and Jeannerod, C. P., Improved backward error bounds for LU and Cholesky factorizations, SIAM. J. Matrix Anal. & Appl., 35[2] 2014, 684–698.
[21] Rump, S. M. and Ogita, T., Super-fast validated solution of linear systems, J. Comput. Appl. Math. 199[2] (2007), 199–206.
[22] Rump, S. M., Ogita, T. and Oishi, S., Accurate floating-point summation part I : faithful rounding, SIAM J. Sci. Comput. 31(2008), 189–224.
[23] Rump, S. M., Ogita, T. and Oishi, S., Accurate floating-point summation part II : sign, K-fold faithful and rounding to nearest, SIAM J. Sci. Comput. 31(2008), 1269–1302.
[24] Stewart, G. W. and Sun, J. G., Matrix Perturbation Theory, Academic Press, Boston, 1990.
[25] 杉原正顯，室田一雄，線形計算の数理，岩波書店，2009.
[26] Yanagisawa, Y., Ogita, T. and Oishi, S., Convergence analysis of an algorithm for accurate inverse Cholesky factorization, Japan Journal of Industrial and Applied Mathematics, 31(2014), 461–482.
[27] Yanagisawa, Y., Ogita, T. and Oishi, S., A modified algorithm for accurate inverse Cholesky factorization, Nonliner Theory and Its Applications, 5[1](2014), 35–46.

[Abstract]

This paper is concerned with an iterative algorithm for an accurate inverse matrix factorization which requires an algorithm for accurate dot product, which helps to treat ill-conditioned matrices. Following the results by Rump[15], Ogita[8], Ogita and Oishi[9] derived an such iterative algorithm. Firstly, we explicate Rump's method[15] for inverting an ill-conditioned matrix. We then focus on the algorithm for an accurate inverse Cholesky factorization via the adaption of Rump's framework directly to shifted Cholesky factorization of symmetric and positive definite matrices. Furthermore, we present some numerical results from a comparison of the algorithm with a standard Cholesky factorization using long precision arithmetic [5, 6], in terms of measured computing time for verifying the positive definiteness of an input matrix.