線形方程式の解の精度保証に対するOishiRump法

2009SE197 中野綾子
指導教員：杉浦洋

1　はじめに

本論文では，Shin'Ichi Oishi, Siegfried M.Rump[1] による線形方程式の高速精度保証付き解法を研究し，その有効性を Mathematica による数値実験で検証する．コンピュータでは，高速に数値計算するために，実数を浮動小数点数で近似して計算を行っている．

本研究では，線形方程式 $A\boldsymbol{x} = \boldsymbol{b}$ の解を精度保証付きで求める方法を考える．最近，効率的な精度保証付き数値計算方法が開発され，元数の大きい方程式に対しても，実用的な精度保証が得られるようになってきた．本研究では，その方法を学び，理解を深め，Mathematica 用の高速な精度保証アルゴリズムの構成を目指す．

2　精度保証の基本定理

［定理 1］方程式 $Ax = b$ の近似解 $\tilde{x}$ と A の近似逆行列 A^+ が与えられたとする．行列 $R = A^+A - I$ が不等式

$$\| R \|_\infty < 1$$

が満たされたなら，真の解 x^* が存在し，

$$\|x^* - \tilde{x}\|_\infty \leq \frac{\|A^+r\|_\infty}{1 - \|R\|_\infty} \leq \frac{\|A^+\|_\infty\|r\|_\infty}{1 - \|R\|_\infty} \qquad (1)$$

が成り立つ．ここで $r = A\tilde{x} - b$ は残差である．//
式 (1) の右辺を精度保証付きで計算することにより，誤差 $\|x^* - \tilde{x}\|_\infty$ の上界が得られる．重要なのは $\| R \|_\infty$ と $\| r \|_\infty$ の限界の計算である．

低速アルゴリズム　　$\| R \|_\infty$ と $\| r \|_\infty$ の限界
$setround(down)$;
$\underline{R} = fl(A^+A - I)$;
$\underline{r} = fl(r)$;
$setround(up)$;
$\overline{R} = fl(A^+A - I)$;
$\overline{r} = fl(r)$;
$r_{mid} = fl(\underline{r} + \overline{r}/2)$;
$r_{rad} = fl(r_{mid} - \underline{r})$;

ここで，$setround(down)$，$setround(up)$ は丸めモードの切り替えである．しかし，丸めモードの切り替え機能は，Mathematica に実装されていないため，数値実験ではこれを等価な区間演算で置き換えた．

3　高速精度保証

式 (1) の右辺の計算において，計算量の大きい部分は逆行列 A^+ の計算と残差 R の計算で，それぞれ計算量は $O(n^3)$ $flops$ である．ここで述べる高速精度保証アルゴリズムの目的は，$\|R\|_\infty$ を直接評価せず事前誤差評価で行い，計算量を $O(n^3)$ $flops$ から $O(n^2)$ $flops$ に削減することである [1].

計算量を削減する方法として，行列 PA の近似 LU 分解 $PA = \tilde{L}\tilde{U}$ と $\tilde{L}$,$\tilde{U}$ の数値逆行列 $\tilde{L}^+ \cong \tilde{L}^{-1}$,$\tilde{U}^+ \cong \tilde{U}^{-1}$ を計算し，定理 1 で $A^+ = P^{-1}\tilde{L}\tilde{U}$ とする．このとき

$$\begin{aligned}\|R\|_\infty &= \|A^+A - I\|_\infty = \|\tilde{U}^+\tilde{L}^+PA - I\|_\infty \\ &= \|\tilde{U}^+\tilde{L}^+(PA - \tilde{L}\tilde{U})\|_\infty \\ &\quad + \|\tilde{U}^+(\tilde{L}^+\tilde{L} - I)\tilde{U}\|_\infty + \|\tilde{U}^+\tilde{U} - I\|_\infty\end{aligned}$$

となる．
第一項は

$$\begin{aligned}&\gamma_n\| \mid \tilde{U}^+ \mid\mid \tilde{L}^+ \mid\mid \tilde{L} \mid\mid \tilde{U} \mid \boldsymbol{e}\|_\infty \\ &+ \delta_n\| \mid \tilde{U}^+ \mid\mid \tilde{L}^+ \mid (n\boldsymbol{e} + \mathrm{diag}(\mid \tilde{U} \mid))\|_\infty \underline{u},\end{aligned}$$

第二項は

$$\begin{aligned}&\gamma_n\| \mid \tilde{U}^+ \mid\mid \tilde{L}^+ \mid\mid \tilde{L} \mid\mid \tilde{U} \mid \boldsymbol{e}\|_\infty \\ &+ n\delta_n\| \mid \tilde{U}^+ \mid \boldsymbol{e}\|_\infty \ \ \| \mid \tilde{U} \mid \boldsymbol{e}\|_\infty \underline{u},\end{aligned}$$

第三項は

$$\gamma_n\| \mid \tilde{U}^+ \mid\mid \tilde{U} \mid \boldsymbol{e}\|_\infty + \delta_n\|n\boldsymbol{e} + \mathrm{diag}(\mid \tilde{U} \mid)\|_\infty \underline{u}$$

でおさえられる．ここで，$\delta_n = n/(1 - nu)$，$\gamma_n = u\delta_n$，$u = 2^{-53}$，$\underline{u} = 2^{-1027}$，$\boldsymbol{e} = (1, \cdots, 1)^T$ である．これが $\|R\|_\infty$ の計算量 $O(n^2)$ $flops$ の事前誤差評価である．

4　Mathematica 用高速精度保証

Mathematica の区間演算は非常に遅い．行列・ベクトルの∞ノルムを事前誤差評価により，区間演算を用いずに求めることを考える．

4.1　ベクトルの 1 ノルム

ベクトル $x = (x_1, x_2, \cdots, x_n)^T \in R^n$ の∞ノルム

$$\| x \|_1 = \sum_{i=1}^{n} \mid x_i \mid$$

の評価を考える．
(定理 1) ベクトル $x \in R^n$ の∞ノルムについて，次の評価式が成り立つ．

$$\| x \|_1 \leq \frac{s}{1 - \gamma_{n-1}} = M = \frac{1}{1 - \gamma_{n-1}} fl[\| x \|_1].$$

アルゴリズムは次のようになる．
1. 機械計算 $fl[\| x \|_\infty]$ を実行.
2. 評価式 (1) の右辺を区間計算し区間上限を $\overline{M}$ とする．
☆定理 1 より $\| x \|_\infty \leq \overline{M}$ である．

表 1 誤差ノルムの厳格な上界

n	上界 (OR 低)	上界 (OR 高)	上界 (M 高速)
2	2.13×10^{-14}	2.13×10^{-14}	1.88×10^{-14}
4	1.24×10^{-13}	1.24×10^{-13}	1.40×10^{-13}
8	5.97×10^{-13}	5.97×10^{-13}	9.04×10^{-13}
16	2.79×10^{-12}	2.79×10^{-12}	6.45×10^{-12}
32	1.52×10^{-11}	1.52×10^{-11}	1.96×10^{-10}
64	1.03×10^{-10}	1.03×10^{-10}	3.53×10^{-9}
128	7.68×10^{-10}	7.68×10^{-10}	7.93×10^{-9}
256		5.92×10^{-9}	6.67×10^{-8}
512			3.01×10^{-7}
1024			2.19×10^{-5}
2048			1.22×10^{-4}

表 2 計算時間

n	時間 (OR 低)	時間 (OR 高)	時間 (M 高速)
2	0	1.60×10^{-2}	2.49×10^{-4}
4	1.50×10^{-2}	1.50×10^{-2}	2.19×10^{-4}
8	1.60×10^{-2}	4.70×10^{-2}	2.96×10^{-4}
16	1.09×10^{-1}	1.09×10^{-1}	2.97×10^{-4}
32	3.43×10^{-1}	2.03×10^{-1}	4.83×10^{-4}
64	2.64	7.64×10^{-1}	1.12×10^{-3}
128	2.07×10^{1}	3.25	3.84×10^{-3}
256		3.50×10^{1}	1.94×10^{-2}
512			1.03×10^{-1}
1024			7.80×10^{-1}
2048			4.66

4.2 行列の∞ノルム

行列 $A = (a_{ij}) \in R^{m \times n}$ の∞ノルム

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

は，上で述べたベクトルの 1-ノルムの評価技法を用いて次のように評価できる．
1. 機械計算 $fl[\| A \|_\infty]$ を実行.
2. 評価式 (7) の最右辺を区間計算し区間上限を $\overline{M}$ とする.
☆ $\| A \|_\infty \le \overline{M}$ が成立する.

4.3 線形方程式の残差ノルム

線形方程式 $Ax = b$, $A = (a_{ij}) \in R^{n \times n}$, $b = (b_i)$, $x = (x_i) \in R^n$ の残差 $r = Ax - b = (r_i) \in R^n$ の∞ノルム $\| r \|_\infty$ の評価も以下のように行える.
1.$| \tilde{r_i} |$,$\tilde{\rho_i}$ を機械計算する.
2.(9) を区間計算し, 区間 $[M_i]$ の上限を $\overline{M_i}$ とする.
3.$M = \max_{1 \le i \le n} \overline{M_i}$ を計算する.
☆ $\| r \|_\infty \le M$ である.

4.4 $\| R \|_\infty$ の上界

$\| R \|_\infty$ の上界も次のように計算できる.
1.$\tilde{R} = fl[A^+ A - I]$ を計算
2.$\tilde{R_1} = fl[| A^+ | \, | A | + I]$ を計算
3.$M_1 = fl[\| \tilde{R} \|_\infty]$ を計算
4.$M_2 = fl[\| \tilde{R_1} \|_\infty]$ を計算
5.$M = \frac{1}{1-\gamma_{n-1}}(M_1 + \frac{\gamma_{n+1}}{1-\gamma_{n+1}} M_2)$ を区間計算.

5 数値実験

三種類のアルゴリズムを Mathematica 上に実現し，数値実験を行った．n 次 Frank 行列を係数行列とする線形方程式の解の精度保証を行った．Oishi，Rump の低速版 (OR 低)，高速版 (OR 高) と今回開発した Mathematica 用のアルゴリズム (M 高速) の近似解の誤差ノルムの厳格な上界を計算した．その結果を表 1，2 に示す．表 1 は，計算された上界の値である．n は方程式の次元である．表 2 は，計算時間を示す.

OR 低と OR 高を比べると誤差ノルム上界は等しい．実行時間では，OR 高の方が短く，比は n が大きくなるほど大きい．n=128 のときは 1/6 となる．今回開発した Mathematica 用のアルゴリズム (M 高速) は，誤差ノルム上界が OR 低，OR 高の約 10 倍大きめにでる．しかし，非常に高速で例えば，n=256 のとき，OR 高に対する実行時間の比が 1/1800 となる．誤差ノルムの上界は，値が小さいほど正確で，大きいほど不正確である．また，時間が短いほど良い.

6 終わりに

本研究では，Shin'Ichi Oishi，Siegfried M.Rump[1] により，線形方程式の解を精度保証付きで求める方法を学び，Mathematica 上でプログラムを実装し数値実験を行った．三輪 [2] は正規数領域で計算が行われると仮定し研究を行った．本研究では，正規数領域と副正規数領域で計算を行われる場合に拡張し，研究を行った．しかし，Mathematica の数システムを調査し，副正規数が使われていないことが数値実験により分かり，副正規数領域の実験は行わなかった．精度保証付き計算では，Oishi，Rump の低速アルゴリズム，高速アルゴリズムを実装し，数値実験を行った．高速アルゴリズムは，方程式の次元が 128 のとき，約 6 倍高速であったが，きわめて遅い．その原因は，Mathematica の区間演算の低速性にある．そこで，事前誤差解析に基づき，Mathematica 専用の高速アルゴリズムを開発した．このアルゴリズムは，誤差ノルムの上界が約 10 倍大きめにでるが，非常に高速で，方程式の次元が 256 のとき，Oishi，Rump の高速版に比べて約 1800 倍高速である.

参考文献

[1] Shin'Ichi Oishi, Siegfried M.Rump:Fast verification of solutions of matrix equations, Numer. Math. Vol. 90, pp. 755-773(2002).

[2] 三輪嵩 :「線形方程式の精度保証付き解法」. 南山大学数理情報学部数理科学科卒業論文 (2010).