線形方程式の精度保証付き解法

2004MM012 後藤佳紀
指導教員: 杉浦洋

1 はじめに

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．したがって，得られる数値計算の結果もまた，近似である．従来はこの数値を検算するのに精度と計算量の問題があったが，多くの研究により，数値計算による近似計算の検算が，精度よく高速にできるようになった．

本研究では，線形方程式$Ax = b$の解を精度保証付きで求める方法を考える．理論上はガウスの消去法を区間演算で実行することによって精度が保証されるが，元数が大きくなるにつれ，解の存在区間が過大になり実用性に欠ける．最近，効率的な精度保証付き数値計算の方法が開発され，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，プログラムを作成し，その特性を数値実験により確認する．

2 精度保証付き計算について

2.1 浮動小数点数とは

浮動小数点数システムの規格として，IEEE754が多くのコンピュータで標準的に用いられている．本研究ではIEEE754に基づく2進数浮動小数点数システムを利用する．

IEEE754による浮動小数点数には4つのタイプが規定されている．規格化2進浮動小数点数，零，非規格化2進浮動小数点数，NaNの4つである．前3つを2進浮動小数点数とよぶ．

2進浮動小数点数とは

$$a = \pm \left(\sum_{k=0}^{N} 2^{-k} d_k \right) \times 2^e,\ d_i \in \{0, 1\}$$

と書ける数である．eは指数という．本論文の数値実験は全て拡張倍精度でおこなった．拡張倍精度では$N = 63$, $-32767 \leq e \leq 32768$となる．拡張倍精度2進浮動小数点数全体の集合を$\mathbb{F}$とする．

2.2 丸めモードについて

IEEE754では集合$\mathbb{F}$上での演算は丸めを用いて計算されており，主に4 つの丸めモードが実装されている．

一つめは上向きの丸め(丸め上げ)で，実数$c \in \mathbb{R}$に対して，c以下の浮動小数点数の中で最も小さい数に丸めるモードである．

二つめは下向きの丸め(丸め下げ)で，c以下の浮動小数点数の中で最も大きい数に丸めるモードである．

三つめは最近点への丸め(丸め込み)で，cに最も近い浮動小数点数に丸めるモードである．

最後に切捨てで，絶対値がc以下の浮動小数点数の中でcに最も近い数に丸めるモードとなる．

四則演算$\cdot \in \{+, -, \times, /\}$の丸めモード$\bigcirc \in \{\triangle, \triangledown, \square\}$における機械演算$\odot$は次のように定義される．

$$x \odot y = \bigcirc(x \cdot y)\ (x, y \in \mathbb{R})$$

2.3 精度保証付き線形計算の基本算法

2.3.1 内積の精度保証

n次元列ベクトル$x = (x_i), y = (y_i)$の内積$z = x \cdot y$の値を精度保証付きで計算する問題を考える．精度保証付き数値計算においては，丸めのモードの指定の切り換えが必要となる．$setround(up)$を上への丸め，$setround(down)$を下への丸めのモードへの切り換え命令とする．$x_i, y_i \in \mathbb{F}$のとき内積の精度保証付き計算は次のようになる．

$$\begin{aligned} &setround(down);\\ &\underline{z} = \sum_{i=1}^{n} x_i \cdot y_i;\\ &setround(up);\\ &\overline{z} = \sum_{i=1}^{n} x_i \cdot y_i; \end{aligned}$$

丸めのモードの切り換えにより，$x_i \times y_i$及び$\sum$の計算において，それぞれ正確な値よりも小さな値，大きな値が得られる．これにより$\underline{z} \leq z \leq \overline{z}$，すなわち$z \in [\underline{z}, \overline{z}]$が数学的に厳密な意味で成立することになる．ここで$n$次元列ベクトル$a, b$において，

$$a \leq b \Leftrightarrow a_i \leq b_i\ (1 \leq i \leq n)$$

である．

2.3.2 行列積の精度保証

次に行列の積を考える．$n \times n$行列A, Bの積$A \times B$をCに格納する関数$mul(A, B, C)$があるとする．この関数が浮動小数点数の加法と乗法のみで計算されるとするとき

$$\begin{aligned} &setround(down);\\ &mul(A, B, \underline{C});\\ &setround(up);\\ &mul(A, B, \overline{C}); \end{aligned}$$

と計算すると，Cの要素計算はAの行ベクトルとBの列ベクトルの内積計算だから，$\underline{C} \leq AB \leq \overline{C}$と計算結果の精度保証ができる．ここで$m \times n$行列$A = (a_{ij}), B = (b_{ij})$について

$$A \leq B\ \Leftrightarrow\ a_{ij} \leq b_{ij}\ (1 \leq i \leq m, 1 \leq j \leq n)$$

とする．

3 線形方程式の近似解について

3.1 精度保証のための定理

定理 1 方程式$Ax = b$の近似解$\tilde{x}$とAの逆行列の近似行列Rが求められたとき，行列$G = RA - I$が不等式

$$||G||_\infty < 1$$

を満たすとき，真の解x^*が存在し，

$$||x^* - \tilde{x}||_\infty \leq \frac{||R(b - A\tilde{x})||_\infty}{1 - ||G||_\infty} \leq \frac{||R||_\infty ||b - A\tilde{x}||_\infty}{1 - ||G||_\infty}$$

が成り立つ．

3.2 計算手順

定理1による真の解x^*の精度保証付き計算手順を以下に示す．

1. 近似解$\tilde{x}$，近似逆行列RをLU分解法により計算する．
2. $H = \max\{|\underline{G}|, |\overline{G}|\}$とし，$M_G = ||H||_\infty$を丸め上げで計算する．これにより，$||G||_\infty \leq M_G$となる．同様にして，$||b - A\tilde{x}||_\infty \leq M_R$，$||R||_\infty \leq M_r$なる上界$M_R, M_r$を計算する．
3. 定理1より

$$||\tilde{x} - x^*||_\infty \leq E_x = \frac{M_R M_r}{1 - M_G}$$

である．この右辺を分子は丸め上げ，分母は丸め下げ，割算は丸め上げで計算する．

4 数値実験

4.1 実験概要

使用した計算機は東芝dinabook CX1/212CEで，CPUはIntel celeron M，1.2GHzである．C言語およびC++で拡張倍精度のプログラムを作成し，Linux上のコンパイラgccでオプティマイズなしでコンパイルし，実行した．

n次元方程式$Ax = b$は，$A = (a_{ij}) \in \mathbb{F}^{n\times n}, b \in \mathbb{F}$とした．$a_{ij}$は$[-1, 1]$の疑似一様乱数，$b_i = \sum_{j=1}^{n} a_{ij}$である．解は$x^* = (1, \cdots, 1)^T$となる．この方程式に対しLU分解法で求めた数値解を$\tilde{x}$とする．これに対し，誤差ノルム$e_x = ||\tilde{x} - x^*||_\infty$と，今回のアルゴリズムで，誤差ノルム$e_x$の上界$E_x$を求めて出力するプログラムを作成した．

実験では，$n = 2^m$ $(3 \leq m \leq 8)$とし，各nにつき1000個の方程式を生成し，計6000題を今回作成したプログラムで解いた．

比較のため，同じ1000個の方程式を区間演算のLU分解法でも解いてみた．

4.2 実験結果

今回の方法では，すべての方程式の解の精度保証に成功した．すなわち，すべての方程式で$e_x \leq E_x$であった．$\log_{10} e_x$，$\log_{10} E_x$の平均を表1に示す．同じ表に，区間LU分解法による$\log_{10} E_x$の値を示す．区間LU 分解法は，$6 \leq m$では，計算途中で0割算が起こり，失敗した．表ではそれを傍線で示した．

表 1: 今回の方法と区間LU分解法の計算結果の平均

	今回の方法		区間LU分解法
n	$\log_{10} e_x$	$\log_{10} E_x$	$\log_{10} E_x$
8	-17.57	-16.25	-14.43
16	-16.98	-15.49	-10.32
32	-16.76	-14.93	-3.09
64	-16.37	-14.38	——
128	-15.44	-13.33	——
256	-14.88	-12.53	——

この表によると，今回の方法でも区間LU分解法でも，行列の次元nが大きくなるにつれて，上界が大きくなっている．これは計算回数が増加するために誤差の蓄積量が増加していくためである．しかし，区間LU分解法における増大は急激で，$n = 64(m = 6)$で計算不能となった．

表 2: 今回の方法と区間LU分解法の平均実行時間(単位：秒)

n	今回の方法	区間LU分解法
8	8.0×10^{-5}	7.9×10^{-4}
16	6.6×10^{-4}	5.1×10^{-3}
32	4.0×10^{-3}	3.7×10^{-2}
64	2.9×10^{-2}	——
128	2.3×10^{-1}	——
256	3.7×10^{1}	——

次に，1つの方程式を解くのに要した時間を表2に示す．区間LU 分解法は，今回の方法より実行時間が多くかかり，非効率的であることがわかる．これは区間演算が通常の演算より遅いことが原因である．今回の方法では，大部分の計算が丸めモードを切替えない通常の演算で行なわれる．これが計算効率に寄与している．

5 おわりに

本研究では，線形方程式の解法のプログラムを実装した．これにより，用意したすべての方程式で解の精度保証に成功した．理論上は区間演算によって精度が保証されるが，今回の方法によって，区間LU分解法よりかなり小さい誤差ノルムの上界を与えることができる．

今回のプログラムでは，メモリの割り付け不良のため，大きな次元nでは実験できなかった．この点では改良を要する．また，実験の簡略化のために，真の解ベクトルを$(1, 1, \cdots, 1, 1)^T$となるようにしたため，他のいかなる解においても実験可能になるようにすることも必要だと思われる．さらに，計算の効率化，高速化に関する研究も必要である．

参考文献

[1] 大石進一：精度保証付き数値計算，コロナ社 (2000).

[2] 杉浦洋：数値計算の基礎と応用-数値解析学への入門-，サイエンス社 (1997).

[3] 大澤智弘：区間演算システムの構築，南山大学数理情報学部数理科学科卒業論文 (2006).