数 値 解 析 31－3
（1989．12．8）

有理数演算を用いた線形方程式の解法の振舞い

大柳俊夫　大内 東
北海道大学工学部

一般に，線形方程式は四則演算のみで計算を行うことが可能で，有理数演算を適用することができる．この演算を用いることにより厳密な計算を行うことが可能であるが，計算時間とメモリー容量に関して問題があり，実用的な方法とは言い難く今までほとんど利用されていなかった．ところが，最近の計算機の性能の向上や数式処理システムの急速な普及などにより，この演算の利用の可能性は次第に高まりつつあると考えられる．

本論文は，線形方程式の代表的な解法であるＬＵ分解による方法に有理数演算を適用した場合の計算の振舞いを調べる目的で行った，３通りのピボット選択を用いた数値実験とその結果について述べるものである．

A Behavior of an Algorithm for Linear Equations using Rational Arithmetic

Toshio Ohyanagi　Azume Ohuchi
Faculty of Engineering,
Hokkaido University

North 13　West 8, North ward, Sapporo 060, Japan

Linear equations are solved by using four arithmetical operations. Therefore rational arithmetic can be used to implement the algorithm for the problems. Since using rational arithmetic, however, requires much computation than using floating-point arithmetic, this arithmetic have not been used widely in the field of numerical computation. Recently some computer languages support this arithmetic in its specification. And also many algebraic computation languages have come into wide use. So it is considered that this arithmetic will become more available in a few years.

This paper intends to investigate a behavior of LU decomposition algorithm implemented by using rational arithmetic. For that purpose, three pivot selection rules are considered and then some computational experiments are made with them.

1．はじめに

一般に，線形方程式は四則演算のみで計算を行うことが可能で，有理数演算つまり分数計算を適用することができる．この演算を用いると，浮動小数点演算で生じる桁落ちや丸め誤差を排除した厳密な計算を行うことが可能で，このため浮動小数点演算の場合に行われている計算精度を上げるためのスケーリング，ピボット選択および解の反復改良の必要はなくなる[1),2)]．このように計算精度の点からは，これ以上精度を改善することはできないという意味において望ましい演算ではあるが，解くことが可能な問題のサイズや計算時間の点で，浮動小数点演算を行うよりも不利であり，あまり実用的な方法とはいえずほとんど利用されていなかった．ところが，最近の計算機のハードウェアの性能の向上は著しく，また有理数演算を仕様としてサポートする計算機言語や数式処理システムの急速な普及などにより，有理数演算の利用の可能性は次第に高まりつつある．しかし，有理数演算を用いた場合の線形方程式の計算の振舞いに関しては，現在までいくつかの報告があるだけで，ほとんど研究は行われていない[3)]．

本論文は，線形方程式の代表的な解法であるＬＵ分解による方法に有理数演算を適用した場合の計算の振舞いを調べることを目的とする．そのために，３通りのピボット選択方法を考え，数値実験により，選択方法の違いによる使用メモリー容量，計算時間などを詳しく調べ，その結果をもとに計算の振舞いを考察する．

以下，第２章で今回の実験で用いた有理数演算方法を説明し，第３章で有理数演算によるＬＵ分解の特徴について述べる．第４章では今回行った実験の内容とその実験結果を示す．そして第５章で，有理数演算を用いたＬＵ分解による線形方程式の計算の振舞いを考察する．

2．有理数演算を行うための準備

2.1 有理数演算を行う方法

現在，コンピュータ上で有理数演算を行う方法は大きく３つあり，第一に有理数演算を言語仕様としてサポートしているプログラミング言語（例えばCOMMON LISP ）を用いる方法，第二に最近注目を集めている数式処理システム（例えばREDUCE, MACSYMA, Mathematicaなど）を用いる方法，そして第三に有理数演算を行うためのライブラリーを独自に作成して用いる方法である．

本論文で行う実験では以下の理由から第三の方法をとることにした．

(1) 使用メモリー容量を詳細に調べるには，第一および第二の方法では困難であると考えられる．

(2) 第一の有理数演算を言語仕様としてサポートしているプログラミング言語はまだそれほど多くなく，数値計算の分野の研究者はほとんど利用していないことが予想される．

(3) 第二の数式処理システムは，一般にインタープリタであるために現在まだ速度の点で問題である．

2.2 有理数演算ライブラリーのインプリメント

有理数演算ライブラリーの記述言語として以下の理由からPascalを選択した．

(1) 数値計算の分野で主流のFORTRAN と比較した場合，Pascalのほうが柔軟なデータ構造が構築でき，プログラム開発の初期段階におけるデータ構造の変更等が容易に行える．また，プログラムのドキュメント性においても優れている．

(2) ISO 規格[4)]に準拠して記述すれば異なる計算機上への移植が容易に行える．

なお，使用したコンパイラはPro Pascal，使用した計算機はPC-9801vx4(CPU:80286,CLOCK:8MHz)とJ3100SGT(CPU:80386, CLOCK:16MHz)である．PC-9801vx4は主にプログラムの開発に使用し，J3100SGTは数値実験で使用した．

図１にライブラリーで用いた主なデータ構造を示す． Ratio型が有理数型で，符号を表すフィールドsignと，分子，分母をそれぞれ表すUsBigNum型のフィールドnum，denからなるレコード構造である．UsBigNum型は符号なし任意多倍長整数型で，

```
(* 定数宣言 *)
  const
     MaxCard = 65535;(* 2^16-1 *)
(*  型宣言  *)
  type
     Cardinal = 0 .. MaxCard;
     CellPtr  = ^Cell;
     Cell     = record
                   Value : Cardinal;
                   Hptr,Lptr : CellPtr
                end;
     UsBigNum = record
                   Cells : Cardinal;
                   HSTptr,LSTptr : CellPtr
                end;
     Ratio    = record
                   sign : -1 .. 1;
                   num,den : UsBigNum
                end;
```

図1　有理数演算のためのデータ構造.

(例) a = 15! = 1307674368000 (10進)

```
65536 )1307674368000          剰余
65536 )     19953527   ...    22528
              304      ...    30583
```

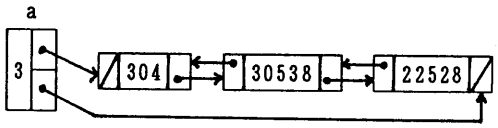


図2　符号なし任意多倍長整数の例.

任意多倍長整数を表すために使用しているセルの数を表す Cellsと，任意多倍長整数の最上位，最下位のセルをそれぞれ指すポインターのフィールドであるHSTptr，LSTptrからなるレコード構造である．Cell型は任意多倍長整数を表す基本単位の型で，そのセルの上位および下位のセルへのポインターのフィールドであるHptr，Lptr，そのセルに格納されている値を表す Valueからなるレコード構造である．このセル1つが，65536（2の16乗）進数の1桁を表すものである．

ここで，図2にUsBigNum型の数の例を示す．図に示すように，基本的に双方向リストを用いている．この理由は，任意多倍長整数の基本演算の中で，加減乗算が最下位の桁からの計算に対し，除算，比較演算が最上位の桁からの計算であるために，双方向からのアクセスを高速にするためである．つまり，メモリーの節約よりも計算速度を重視した設計となっている．

そして，UsBigNum型に対する四則演算，比較演算を行う基本手続きを文献5)に従ってインプリメントし，それをもとにしてRatio 型に対する同様の基本手続きを用意して有理数演算ライブラリーとした．

3．有理数演算によるＬＵ分解

一般に，線形方程式は，

$$Ax = b \qquad (1)$$

という行列形式で表現される．ここでＡはｎ次の正方行列，ｘ，ｂはｎ次の列ベクトルとする．この問題の代表的な直接解法にＬＵ分解を行う方法がある[1),2)]．この方法は，行列Ａを下三角行列Ｌと上三角行列Ｕの積の形に分解して(1) 式を，

$$LUx = b \qquad (2)$$

という形式にし，その後で，

$$Ly = b,\ Ux = y \qquad (3)$$

という２組の方程式を解くものである．(3) 式の方程式は容易に解くことができる．

この方法を浮動小数点演算を用いてインプリメントする場合，計算精度を上げるために，前処理として行列Ａのスケーリング，ＬＵ分解の際のピボット選択，そして解が得られた後の解の反復改良などが行われる．これに対して有理数演算を用いる場合は，浮動小数点演算の場合に生じるような桁落ちや丸め誤差は生じないので計算精度は保証される．従って，計算精度を上げる目的で浮動小数点演算の場合に行わなければならかった処理は必要がなくなる．その反面，浮動小数点演算の

場合に比べ，一般に必要なメモリー容量が多くなることと，計算時間が長くなるという問題が生じる．これは，浮動小数点演算の場合はある浮動小数点数を表すために必要なメモリー容量が計算途中一定であのに対し，有理数演算の場合は計算途中で分子分母の中間膨張が起こることがあり，多くのメモリーと多くの計算時間が必要になるためである．このことから，有理数演算を高速に行うためには，計算途中での分子分母の中間膨張を抑え，使用するメモリー容量をできる限り少なくする方法を検討する必要がある．そのため本論文では，３通りのピボット選択方法を考え，数値実験により，選択方法の違いによる使用メモリー容量，計算時間などを詳しく調べる．そして，その結果をもとに計算の振舞いを検討する．今回考える３つのピボット選択は，行列Ａの第ｋ列成分をa_{ik}とすると，以下のｐ に対応する要素a_{pk} を第ｋ段階目の分解のピボットとするものである．

Ｐ１：$p = \min_{k \leq i \leq n} \{i \mid a_{ik} \neq 0\}$.

Ｐ２：$|a_{pk}| = \max_{k \leq i \leq n} \{|a_{ik}| \mid a_{ik} \neq 0\}$.

Ｐ３：a_{ik}の分子，分母のセル数をそれぞれCn_i，Cd_iとしたとき，

$Cn_p + Cd_p = \min_{k \leq i \leq n} \{Cn_i + Cd_i \mid a_{ik} \neq 0\}$.

Ｐ１はピボット列の中で最初の非零要素をピボットとするもので，ピボット選択としては最も単純なものである．Ｐ２はピボット列の中の非零要素の中で絶対値最大の要素をピボットとするもので，浮動小数点演算を用いる場合に通常行われている部分的ピボッティングである．そして，Ｐ３はピボット列の中で分子分母の桁数（65536進数）の和が最小の要素をピボットとするもので，これはＬＵ分解による分子分母の数の中間膨張をできる限り抑えることを目的とするものである．なお，使用メモリー容量は，使用セル数を調べることで知ることが可能なので，実験では使用セル数を調べる．

４．数値実験

4.1 テスト問題

有理数演算を行う場合，行列Ａの構造および各要素の値により，使用メモリー容量や計算時間が影響されることが予想される．今回の実験では，行列Ａを帯幅５の帯行列に限定し，その要素a_{ij}として以下の４つの形式を考える．

①data-1: $1 \leq a_{ij} \leq 5$ の範囲の整数乱数
$(1 \leq i, j \leq n, |i-j| \leq 2)$

②data-2: $1 \leq a_{ij} \leq 20$の範囲の整数乱数
$(1 \leq i, j \leq n, |i-j| \leq 2)$

③data-3: $1 \leq a_{ij} \leq 10000$ の範囲でランダムに発生させた素数$(1 \leq i, j \leq n, |i-j| \leq 2)$
ただし，全ての値は異なるようにする．

④data-4: m_{ij}を$-5 \leq m_{ij} \leq 5$ の範囲の整数乱数$(1 \leq i, j \leq n, |i-j| \leq 1)$とするとき，

$$a_{ij} = \sum_{k=1}^{n} m_{ik} \cdot m_{jk} \quad (1 \leq i, j \leq n)$$

ここで，data-1，data-2，data-4に対してはＡの次数ｎを30から10きざみに100 までとし，またdata-3に対してはｎを10から10きざみに50までとして，それぞれのｎに対して乱数の種を変えて５題の問題を作成する．

また，右辺列ベクトルｂの成分b_i は，

$$b_i = \sum_{j=1}^{n} a_{ij} \quad (1 \leq i \leq n)$$

とした．

4.2 実験の内容

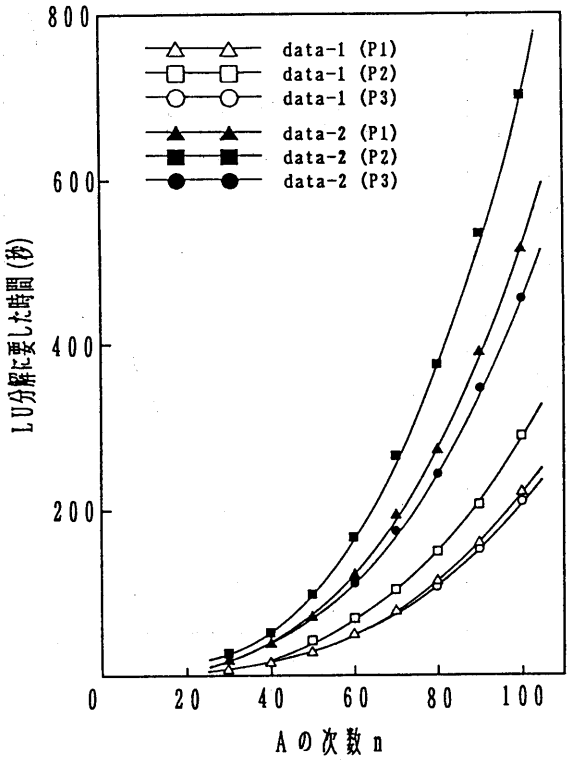
３つのピボット選択方法を用いて４つの形式のテスト問題すべてを解き，以下のことを調べる．

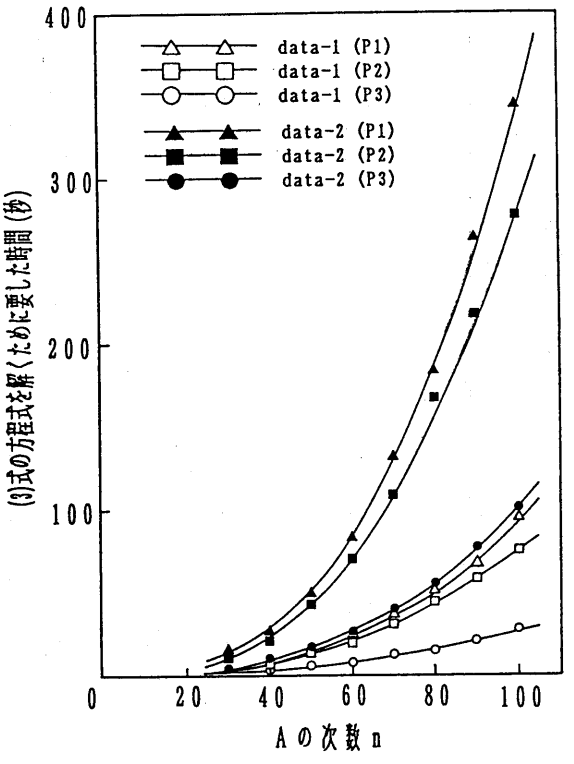
①線形方程式の計算時間
＝ＬＵ分解に要する時間＋
(3) 式を解くために要する時間

②行列Ａの非零要素数とＬＵ分解後の行列Ｌ，Ｕの非零要素数

③行列Ｌ，Ｕで使用しているセルの総数

④行列Ｌ，Ｕ中の非零要素１個あたりの使用セル数の平均と分散





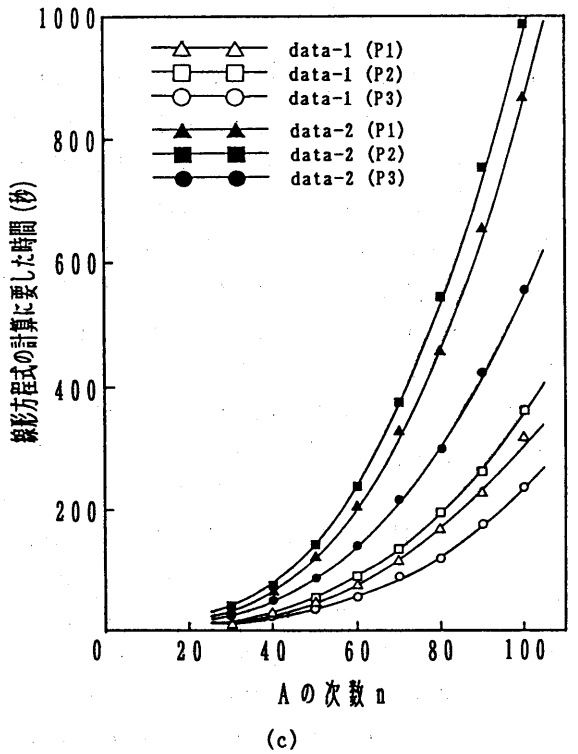


図3．計算時間測定結果(data-1, data-2)

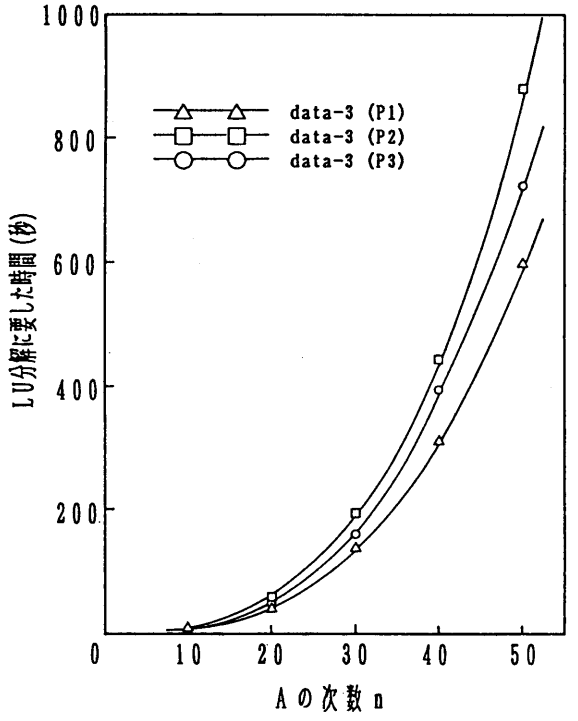
⑤ＬＵ分解の間に行う有理数の既約化の回数
⑥全計算で必要とするセル数
⑦(3) 式を解く間に行う有理数の既約化の回数

なお，行列ＡをＬＵ分解する方法は，今回の実験では外積形式ガウス法とした[2]．

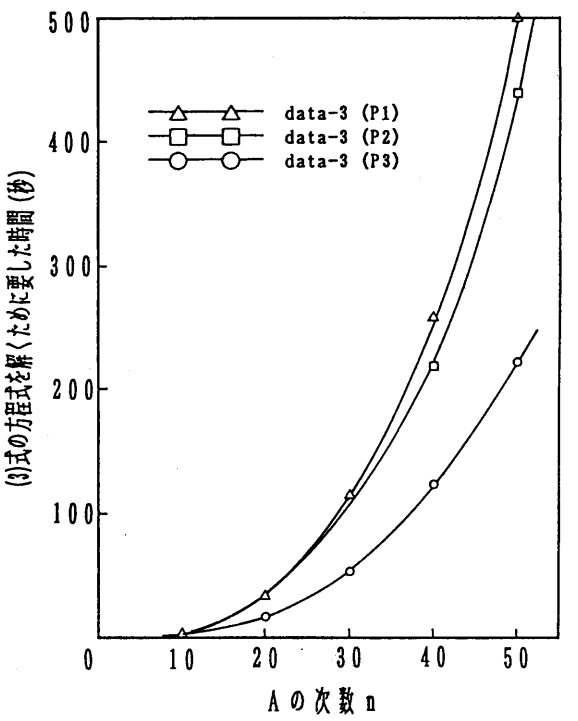
4.3 実験結果

図３－(a)，(b)にdata-1とdata-2に対するＬＵ分解および(3) 式の方程式を解くために要した時間の測定結果を示す．また，図３－(c) にそれらの時間を合計した線形方程式の計算時間を示す．なお，グラフ上の点は，問題を５題解いて得られた結果の平均値である．また，図４－(a)，(b)，(c) にdata-3に対する同様の測定結果，図５にdata-4に対する線形方程式の計算時間の測定結果を示す．

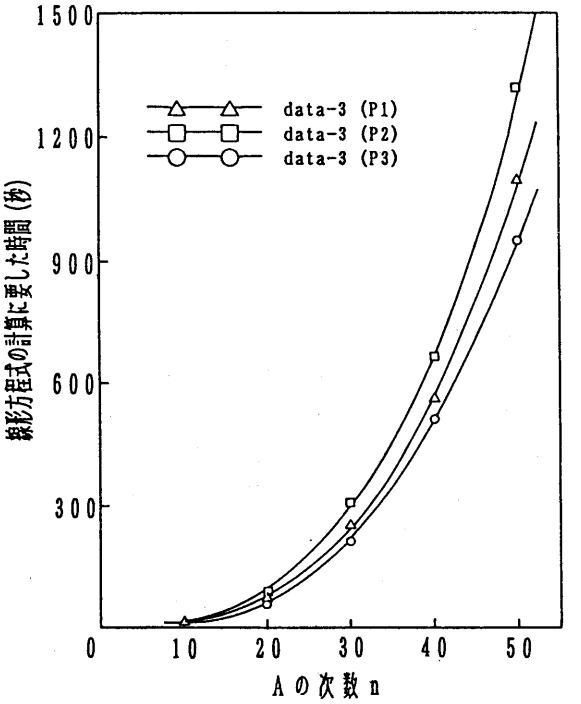
また，行列の次数ｎが50の場合の実験項目①から⑦までを調べた結果を表１に示す．表中でＰ１，Ｐ２，Ｐ３はそれぞれピボット選択方法を示し，



(a)



(b)



(c)

図４．計算時間測定結果(data-3)

data-1, data-2, data-3, data-4はテスト問題の形式を示す．①行は上段がＬＵ分解に要した時間，中段が(3) 式の方程式を解くために要した時間，下段が線形方程式の計算時間である．②行は上段がＬＵ分解後の行列Ｌ，Ｕの非零要素数，下段が行列Ａの非零要素数である．③行は行列Ｌ，Ｕで使用しているセルの総数，④行は上段が行列Ｌ，Ｕ中の非零要素１個あたりの使用セル数の平均，下段がその分散である．そして⑤行は上段がＬＵ分解の間に実際に有理数の既約化を行った回数，下段が有理数の既約化の可能性を調べた回数を表す．また，⑥行は全計算で必要とするセル数，⑦行は上段が(3) 式を解く間に実際に有理数の既約化を行った回数，下段が有理数の既約化の可能性を調べた回数を表す．なお表中の値は，問題５題に対して得られた結果の平均値を，実験項目②，③，⑤，⑥，⑦では小数点以下第１位で，①，④では小数点以下第２位で四捨五入したものである．

この結果，ＬＵ分解に要する時間に関して次のことがわかる．

・行列の次数が等しく構造がほぼ同じでも問題の形式によって時間が大きく異なる．
・問題の形式によって，Ｐ１を用いる方がＰ３を用いるよりも時間が短くなる場合やその逆の場合が生じる．
・Ｐ２を用いる方がＰ１，Ｐ３を用いるよりもＬＵ分解に要する時間が長くなっている．

また，(3) 式の方程式を解くために要した時間に関しても次のことがわかる．

・ＬＵ分解に要する時間の場合と同様に，問題の形式データによって時間が大きく異なる．
・問題の形式によらず次数が同じであれば，Ｐ３を用いる場合が他の場合の半分以下になっている．
・どの問題の形式に対してもＰ３，Ｐ２，Ｐ１の順に時間が短くなっている．

そして，これら２つの時間の和である線形方程式の計算時間に関して次のことが明らかである．

・行列の次数が等しく構造がほぼ同じでも問題の形式によって時間が大きく異なる．

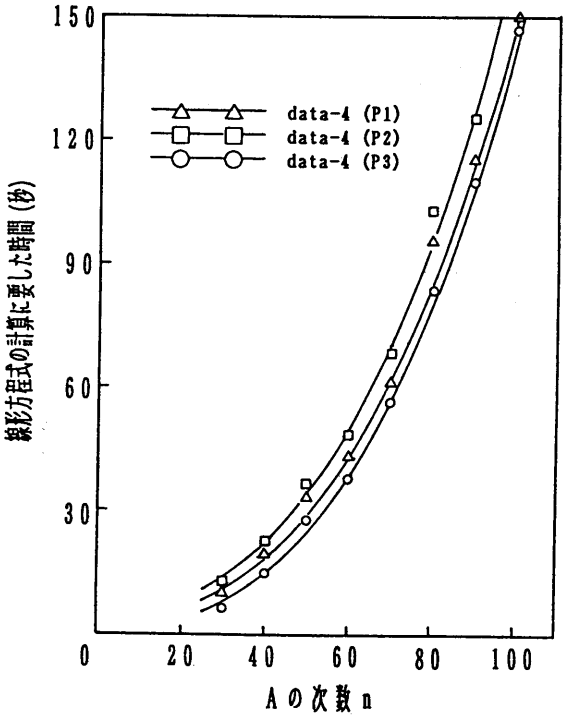


図５．計算時間測定結果(data-4)

表１．各実験項目に関する測定結果（ｎ＝50）

	Ｐ１				Ｐ２				Ｐ３			
項 目	data-1	data-2	data-3	data-4	data-1	data-2	data-3	data-4	data-1	data-2	data-3	data-4
①	33.0	72.4	597.2	24.0	41.6	98.8	877.2	27.4	31.2	70.2	725.2	23.6
	16.8	50.0	499.6	9.2	13.6	43.2	439.8	9.0	6.0	17.2	223.8	4.2
	49.8	122.4	1096.8	33.2	55.2	142.0	1317.0	36.4	37.2	87.4	949.0	27.8
②	244	244	244	227	317	311	311	263	313	327	335	264
	244	244	244	226	244	244	244	226	244	244	244	226
③	1201	2206	7568	836	1221	2161	7197	883	904	1399	4281	682
④	4.9	9.0	31.0	3.7	3.8	7.0	23.2	3.4	2.9	4.3	12.8	2.6
	8.0	37.8	575.0	4.8	6.5	35.3	589.5	4.5	3.0	19.6	404.8	2.0
⑤	186	202	164	176	409	408	282	289	442	555	279	309
	674	677	677	601	892	878	877	699	881	927	950	702
⑥	1731	3144	10667	1247	1647	2905	9503	1265	1239	1903	5794	975
⑦	207	212	214	191	230	237	221	209	194	209	148	195
	436	438	438	400	584	572	571	476	575	604	620	474

・どの問題の形式に対してもＰ３，Ｐ１，Ｐ２の順で計算時間が短くなっていて，その差は次数の増加とともに大きくなる傾向がある．

また表１の結果より，問題の形式によらず以下のことがわかる．

・Ｐ１を用いる場合はＬＵ分解後の非零要素数の増加がほとんどなく，いわゆるフィル・インはほとんど生じていないと考えられる．これに対しＰ２，Ｐ３を用いる場合はＬＵ分解後に非零要素数の増加が起こり，フィル・インが生じている．

・行列Ｌ，Ｕで使用しているセル数はＰ３を用いる場合が最も少なく，Ｐ１，Ｐ２を用いる場合は両者の間でほとんど差はない．

・行列Ｌ，Ｕ中の非零要素１個あたりの使用セル数の平均はＰ３を用いる場合が最小で，次いでＰ２，Ｐ１の順に少なくなっている．また分散に関しても同様の結果となっている．

・ＬＵ分解の計算中の有理数の既約化の割合は，Ｐ１を用いる場合が他の２つの場合に比べて小さくなっいる．Ｐ２，Ｐ３を用いる場合は両者の間でほとんど差はない．

・全計算で必要とするセル数は，行列Ｌ，Ｕで使用しているセル数を調べた結果と同様である．

・(3) 式を解く計算の間の有理数の既約化の割合はＰ１を用いる場合が最大で，次いでＰ２，Ｐ３の順になっている．

５．考察

前章で示した実験結果から，有理数演算を用いた線形方程式の計算の振舞いとして以下のことがあげられる．

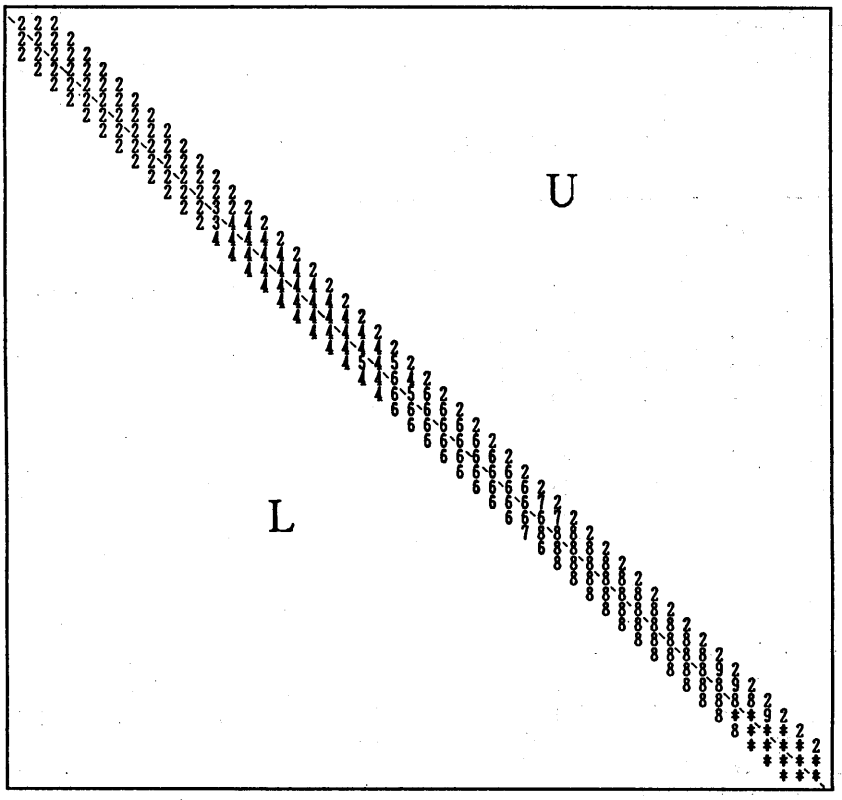


図６－(a)．Ｐ１を用いた場合のＬＵ分解後の非零要素の分布の様子と各非零要素の分子分母の桁数の和

(1) 線形方程式を解くために要する計算時間と使用メモリー容量は問題の形式に大きく影響される．
(2) ピボット選択の違いが使用メモリー容量へ及ぼす影響は大きく，そのため，ＬＵ分解後の非零要素１個あたりの使用セル数の平均と分散も，大きく影響される．
(3) フィル・インの多く生じるピボット選択（Ｐ３）の方が，フィル・インのほとんど生じないピボット選択（Ｐ１）よりも使用メモリー容量と計算時間の点で有利な場合がある．

(1)，(2)は，実験を行う以前にある程度は予想できたことである．しかし，(3) は予想していなかったことであり，有理数演算を用いる場合の特徴的な振舞いの１つと考えられる．そこで，このような現象の生じる原因を調べる実験を行った．実験ではピボット選択Ｐ１とＰ３をそれぞれ用いて，ｎ＝50のdata-1の形式の問題のＬＵ分解後の非零要素の分布の様子と各非零要素の分子分母の桁数の和を調べた．結果を図６の(a)と(b)に示す．(a) はＰ１を用いた場合で，(b) はＰ３を用いた場合である．図中の数は，非零要素の存在する位置と非零要素の分子分母の桁数の和を示す．なお，'＊'は桁数の和が10以上であることを表す．この結果，フィル・インが起こらない場合は，ＬＵ分解が進むにつれ，帯幅４（＝半帯幅×２）の間でしだいに桁数の和が大きくなってきていることがわかる．これに対してフィル・インが起こる場合は，そのフィル・インの大部分が49行と50行の２（＝半帯幅）行で起こっており，ＬＵ分解が進むにつれて，その行の要素の桁数の和のみが大きくなっている．さらにその大きさの変化はＰ１を用いる場合よりも小さくなっている．このように，Ｐ３を用いると，もともと非零である要素の大部分は桁数の和は変化せず，このためＰ１を用いる

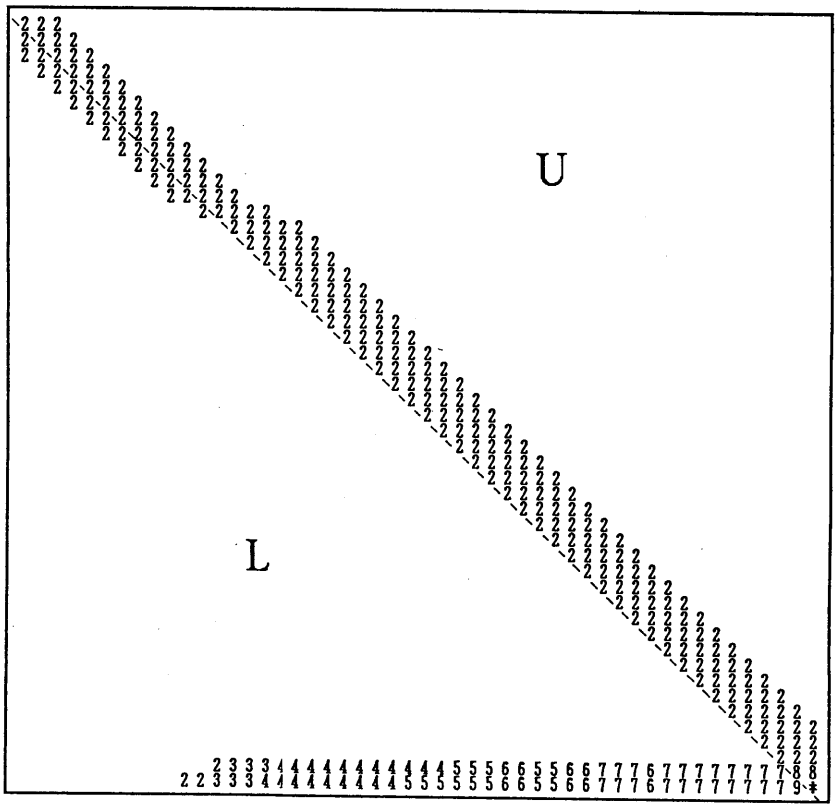


図６－(b)．Ｐ３を用いた場合のＬＵ分解後の非零要素の分布の様子と各非零要素の分子分母の桁数の和

よりも使用メモリーは少なくなる．また，非零要素の数は増えて計算回数が増加しても，その桁数は小さいので計算時間でも有利になると考えられる．

6．おわりに

本論文では，線形方程式の代表的な解法であるＬＵ分解による方法に有理数演算を適用した場合の計算の振舞いを調べるために，３通りのピボット選択方法を考え，数値実験を行って，選択方法の違いによる使用メモリー容量，計算時間などを詳しく調べた．そして，その結果をもとに有理数演算を用いたＬＵ分解による線形方程式の計算の振舞いを考察した．その結果，以下のことが明かとなった．

(1) 線形方程式を解くために要する計算時間と使用メモリー容量は問題の形式に大きく影響される．

(2) ピボット選択の違いが使用メモリー容量へ及ぼす影響は大きく，そのため，ＬＵ分解後の非零要素１個あたりの使用セル数の平均と分散も，大きく影響される．

(3) フィル・インの多く生じるピボット選択の方が，フィル・インのほとんど生じないピボット選択よりも使用メモリー容量の点で有利な場合がある．

今後の課題として，他のピボット選択方法の検討，帯行列以外の形式の問題の場合の計算の振舞いの検討などがあげられる．

参考文献

1)森，名取，鳥居：数値計算，岩波書店，東京(1982)．

2)津田：数値処理プログラミング，岩波書店，東京(1988)．

3)大柳，大内：“有理数演算を用いた連立一次方程式の計算”，第15回システムシンポジウム・第10回知識工学シンポジウム合同シンポジウム講演論文集，p209-214，札幌(1989)

4)原田訳：Pascal(第3版)，培風館，東京(1988)．

5)中川訳：準数値算法/算術演算，サイエンス社，東京(1987)．