授業を補助するプログラム (1)

—行 列 の 演 算—

社会情報学科　行　方　常　幸

1．はじめに

私が担当する数理的な科目（線形代数を扱う「計画数学」等）を扱う授業では，その理解を容易にするために，多くの計算問題を実際に解くことが必須であるものが多い。しかしながら，計算問題を実際に解かせる演習の時間を十分に確保することは，授業時間の制約から，困難であり，演習不足により，授業内容の理解が十分でない状況である。また，テキストのページ数の制限から，解答のついた問題があまり多くなく，自分で演習を行いたい学生も，十分に演習が出来ない状況にある。

このような状況を抜け出すために，よりよい自習環境を提供しようとするのが本稿の目的である。自習環境として求められる要件として，少なくとも，次の3つがあげられる：

(1)　自分が手計算で解ける程度の演習問題であること。

(2)　自分で求めた答えが正解であるかどうかチェックできること。

(3)　自作問題でも演習できること。

行列演算の自習のためにこの3つの要件を満たすプログラムを作成したので本稿で紹介する。

2.「行列の演算」プログラム

作成した「行列の演算」プログラムを起動したのが図1である。この図が示すようにグラフィカルなインターフェースを持つアプリケーションである。

この図から容易に想像できるように，左上のボタンを押し，行列を作成し，その行列式，逆行列，固有多項式等を計算できるプログラムである。行列の固有多項式を求め，連立1次方程式を実際に解いて，この行列の演算プログラムがどのように動作するかを見てみる。

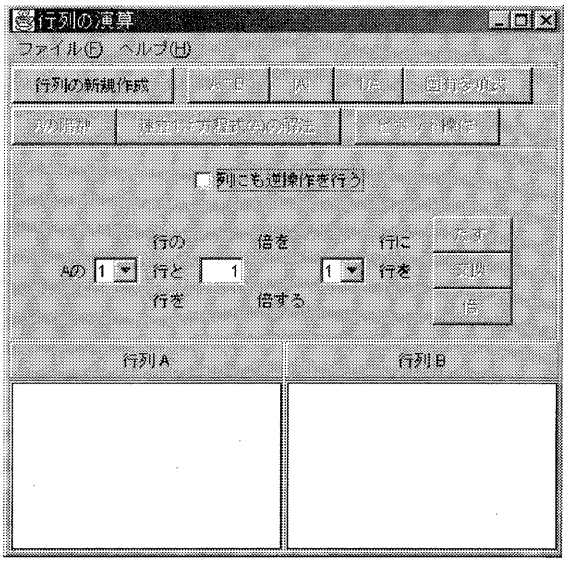


図1　「行列の演算」を起動したところ

まず，次の行列の固有多項式を求める：
$$\begin{pmatrix} 2 & 0 & 0 & 1 & 1 \\ 0 & 2 & 0 & 0 & 0 \\ 3 & 2 & 0 & 0 & 0 \\ 0 & -1 & 0 & 3 & 0 \\ -1 & 1 & 1 & 0 & 0 \end{pmatrix}$$

図1の左上にある「行列の新規作成」ボタンを押すと，図2のように4行4列の行列がウィンドウとしてデスクトップに現れる。求めたいのは5行5列の行列の固有多項式であるから，行数と列数を5に変更して，「サイズの変更」ボタンを押す。そして，上記のデータを入力したのが，図3の行列M1である。

図2　新規に作成された行列 M1

図3　データ入力後の行列

次に，メインのウィンドウ「行列の演算」に戻る（図4参照)。ここで，左下の「行列A」のリストボックスで，行列M1が選択されていること，更に，行列式（|A|)，逆行列（1/A)，固有多項式，等を求めるボタンが利用可能になっていることに注意する。

今，求めたいのは固有多項式であるので「固有多項式」ボタンを押すと，固有多項式を容易に求めることが可能な行列M1-CharFunが新しいウィンドウに表示される（図5参照)。この行列により，答えは

$$(\lambda-2)(\lambda-3)(\lambda^3-2\lambda^2+\lambda-3)$$

である。

さて，次の連立1次方程式を解いてみる。

$$\begin{cases} x_3 + 12x_4 - x_5 = 33 \\ -6x_1 + 3x_2 - 3x_4 + 2x_5 = -63 \\ 5x_1 - 2x_2 + 2x_4 - x_5 = 52 \\ 3x_1 - 3x_3 - 12x_4 + 2x_5 = -3 \end{cases}$$

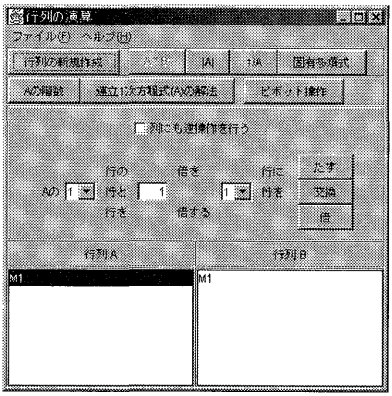


図4　行列AにM1が指定されている

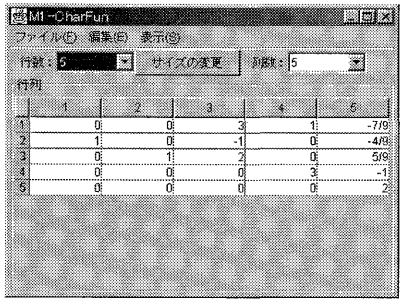
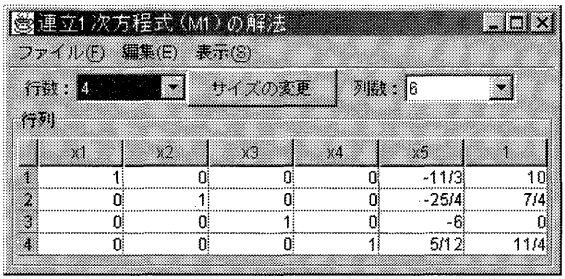


図5　M1-CharFun

デスクトップから不要なウィンドウを閉じるために，行列 M1-CharFun の右上の閉じるボタンを押す。上記のデータは 4 行 6 列だから，図 6 のように行列 M1のサイズを変更し，データも修正する。「行列の演算」ウィンドウに戻って，左下のリストボックスで，行列 A が M1になっていることを確認後，「連立 1 次方程式（A）の解法」ボタンを押すと，解が容易に求められる行列が得られる（図 7 参照)。

	1	2	3	4	5	6
1	0	0	1	12	-1	33
2	-6	3	0	-3	2	-63
3	5	-2	0	2	-5	52
4	3	0	-3	-12	2	-3

図 6　修正後の行列 M1



	x1	x2	x3	x4	x5	1
1	1	0	0	0	-11/3	10
2	0	1	0	0	-25/4	7/4
3	0	0	1	0	-6	0
4	0	0	0	1	5/12	11/4

図 7　連立 1 次方程式の解

「表示」メニューから「解の表示」を選ぶと，図8のように解が陽に表示される。図8において数値が分数表示，小数表示，帯分数表示されているが，この中のどの表示を使うかは「表示」メニューから選択する。

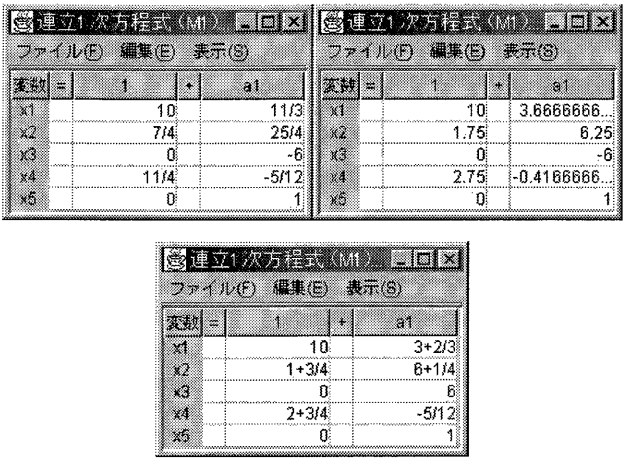


図8　解の表示

このようなGUI（Graphical User Interface）を持つプログラムを作成するために利用できるプログラミング言語としては，Basic，C++，Pascal，Java言語等がある。

私はその中のJava言語を利用した。その第一の理由は，一度プログラムを書けば，どのOSにおいても何も変更せずに実行させることが出来るからである。すなわち，ソースプログラムを書き，コンパイルしクラスファイルを作れば，そのクラスファイルはWindows上であろうと，UNIX上であろうと，（Javaが実行できる環境であれば）そのまま実行できる。Javaを利用した第二の理由は，「はじめに」で述べた目的を，Javaを駆使して，どれくらい容易で安価に実現できるかを見極めることであった。私が利用したのはBorland社製のJBuilder5 Professional版（JBuilder4を利用して作成している間にJBuilde5にバージョンアップされた）である。

Java はオブジェクト指向の言語なので，私の目的に必要なクラスを作成し，その中で，プロパティ，メソッドを定義し，更に，マウスクリック等のイベントを処理するメソッドを追加することがプログラミングの中心である。

3．プログラムの構成

この節では，「行列の演習」プログラムの構成の概略を述べる。

まず，私が意図し最終的に作成した UI（ユーザーインターフェース）は次の通りである。

(A) 行列を Excel にあるようなテーブルで表し，そのテーブルのセルに直接数値を代入する。その時，行列の大きさは指定できるようにして，それ以外の部分は見せない。また，計算の過程を後で調べられるように，各行列を 1 つのウィンドウとして表示する。

(B) 近似計算ではなく可能な限り正確な計算をさせるために，扱える数値は分数とする。数値の表示は，分数，小数，帯分数の 3 つが指定できるようにする。

(C) この分数の入力を処理する際に，明らかな入力ミスは受け付けない。すなわち，「0」から「9」までの数字，小数点「.」，マイナス記号「−」，分数の「/」以外は入力として受け付けない。更に，「.」や「/」は 2 個以上入力できない。「−」は第 2 文字目以降として受け付けない。分数として解釈できない入力に対してはその旨表示し，「0」と解釈する。

(D) 連立 1 次方程式の解は陽に表示できるようにする。

次に，これらをどのように実現したかについて述べる。作成したパブリック・クラスを表 1 にまとめてある。各パブリック・クラス内では多くのプライベート・クラスを利用しているが，その説明は省略する。

まず，(B)の分数を扱う部分が Frac クラスである。このクラスでは，分数を，既約分数として，その分子と分母を保持し，加減乗除最大最小の計算をし，入力文字列から分数を求め，出力として表示用の文字列を作成する。ここで工夫

表1　作成したおもなクラス

クラス名	内　　容
Frac	分数
FracField	Frac の入力用テキストフィールド
FracCellEditor	テーブルで Frac を入力，編集時に利用するエディター
FracCellRenderer	テーブルで Frac を分数，小数，帯分数で表示する
FracMatrix	Frac を要素に持つ行列
VisibleFracMatrix	FracMatrix を表示するフレームウィンドウ
LinearEquSolution	連立1次方程式の解を表示するウィンドウ
MyAboutDialog	バージョン情報を表示するダイアログボックス
MyComboBox	行数，列数，等を設定するコンボボックス
MatrixFrame	「行列の演算」のメインウィンドウ
MyListCellRenderer	MatrixFrame の下方にあるリストボックスの内容を表示する
Application1	main 関数を含む

した点は分子と分母を保持するために BigInteger クラスを使用したこと，Frac クラスに保持されている分数を文字列として出力する際に分数表示，小数表示，帯分数表示出来るようにしたことである。分子と分母を保持するために long を利用することも考えられるが，分数の累乗をすればすぐに桁あふれが生じるため，BigInteger クラスを利用した。BigInteger クラスは Java にあらかじめ定義されている整数を扱うクラスで，記憶すべき数値を正確に保持するためにその都度必要な桁数を確保する。

(C)を実現するのが FracField クラスである。このクラスは Java にあらかじめ定義されている1行のテキストを入力するための JTextField クラスを派生させ，(C)で述べた機能を持たせたものである。

(A)を実現するのが FracMatrix クラス，VisibleFracMatrix クラス，Frac-

CellEditor クラス，FracCellRenderer クラス，MyComboBox クラスである。まず，FracMatrix は与えられた行数と列数からなり Frac を要素に持つ行列である。この FracMatrix クラスで行列の加減乗の演算をし，固有多項式，行列式，逆行列，階数を求め，連立 1 次方程式の解法等を行っている。この FracMatrix を目に見えるようにし，入出力を可能にしたものが VisibleFracMatrix である。図 2 にあるフレームウィンドウが VisibleFracMatrix である。テーブルの利用法に焦点を当てて，行列の入力表示方法を，次節で少し詳しく述べる。

(D)を実現するのが LinearEquSolution クラスで，図 8 のフレームウィンドウである。

表 1 にある他のクラスについて，簡単に説明する。MyAboutDialog クラスは「ヘルプ」メニューから「バージョン情報...」を選んだ時に，表示される図 9 のダイアログボックスである。

MatrixFrame クラスは図 1 のメインウィンドウ「行列の演算」である。この MatrixFrame クラスは新規の行列を作成し，デスクトップ上にあるすべての行列 (VisibleFracMatrix) を下方の 2 つのリストボックス (行列Aと行列B) に列挙し，行列Aのリストボックスで選択された行列を，その時の行列Aとみなし，行列Bのリストボックスで選択された行列が存在すれば，それをその時

図 9　バージョン情報

の行列Bとみなす。行列Aや行列Bが指定されていれば，可能な演算ボタンが利用できるようになる。例えば，行列AとBが両方とも指定されていなければ，「A＊B」ボタンは利用できない。演算ボタンが押されればその演算を行い，結果を表示する。MyListCellRenderer クラスが行列AとBのリストボックスを表示する時に利用されている。すなわち，このリストボックスの内容はVisibleFracMatrix オブジェクトであるが，そのタイトルをリストボックスに表示する。

Application1クラスにはこのプログラムの最初の実行ポイントである main 関数が定義されている。この main 関数が最終的に図1のメインウィンドウ「行列の演算」を表示する。

4．テーブルに利用について

この節では，VisibleFracMatrix クラスのユーザーインターフェースについて説明する。便宜のため，図2を図10として再掲する。

VisibleFracMatrix は JFrame クラスのサブクラスである。JFrame クラス

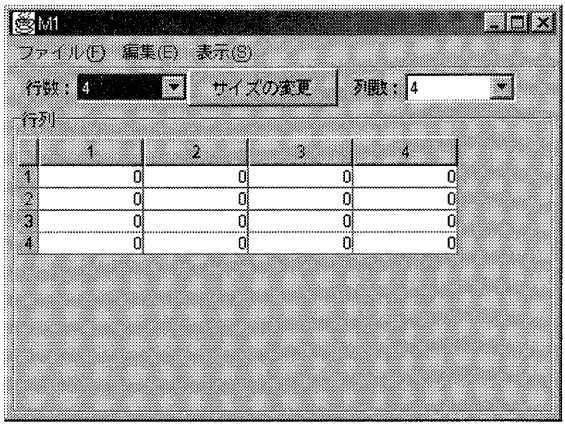


図10　VisibleFracMatrix

はJavaにあらかじめ準備されているクラスで，これを派生させるだけで，最小化，最大化，終了ボタンがあるタイトルバーを持つ，通常のフレームウィンドウが作成できる。このフレームウィンドウに，上のほうから，まず，メニューバーを加え，次に，行数と列数を設定できるMyComboBox，サイズ変更ボタンを加え，最後に行列表示用にテーブルJTableを加える。JTableはJavaにあらかじめ準備されているテーブルである。以上が，VisibleFracMatrixのUIの概略である。

次に，行列の入出力部分であるJTableについて少し詳しく述べる。JTableはヘッダ行とデータの部分で構成されている。図10の列番号（空白，1，2，3，4）が書いてある行がヘッダ行であり，残りの4行（1，0，0，0，0；2，0，0，0，0；3，0，0，0，0；4，0，0，0，0）がデータである。すなわち，行番号（1，2，3，4）をデータとして扱わなければならない。このデータの部分をユーザーに入力させ，入力に応じて表示すれば，われわれの目的を達成することになる。これを行うには，テーブルモデル，セルエディタ，セルレンダラーというものを利用する。

テーブルモデル（MyTableModelクラス）では次のような作業を行っている（表2参照）。各列のクラスを返す（この列のセルの編集時に，このクラスのセルエディタを利用する）；テーブルのデータ部分の現在の行数と列数を返す（これに従って，表示されているテーブルの行数や列数が変化する）；ヘッダ行を表示するための文字列を返す；テーブルのデータ部分の第i行j列の現在の値を返す，及び，新しい値に設定する（この返す元，設定される値の先がFracMatrixの対応する要素（Fracオブジェクト）である）；テーブルのデータ部分のセルが編集可能かどうかを返す。以上のように，JTableは，テーブルモデルによって，表示する内容の受け渡しを行う。そして，われわれの場合この受け渡し内容はFracオブジェクトである。

セルエディタ（FracCellEditorクラス）はユーザーからの入力を処理する。ユーザーが入力する時に，前述のFracFieldオブジェクトを表示し，入力が終われば，このFracFieldオブジェクトの値をJTableに渡す。

セルレンダラー（FracCellRenderer クラス）はテーブルのセルの内容を表示する。FracCellRenderer は Frac オブジェクトを分数表示するか，小数表示するか，帯分数表示するかによって，適切な文字列を生成し，JTable に渡す。このセルレンダラーの存在により，データの内容（Frac オブジェクト）とユー

表2　テーブルモデル

```
public class MyTableModel extends AbstractTableModel {
    public MyTableModel() {
        // コンストラクタ
    }
    public Class getColumnClass(int c) {
        //列 c のクラスを返す
    }
    public int getColumnCount() {
        //現在の列数を返す
    }
    public int getRowCount() {
        //現在の行数を返す
    }
    public String getColumnName(int col) {
        // 列 col の列名（ヘッダ）を返す
    }
    public Object getValueAt(int row, int col) {
        // テーブルの第 row 行 第 col 列の要素を返す。
    }
    public void setValueAt(Object value, int row, int col) {
        // テーブルの第 row 行 第 col 列の要素を value に設定する
    }
    public boolean isCellEditable(int row, int col) {
        // テーブルの第 row 行 第 col 列が
        // 編集可能ならば true を不可能ならば false を返す
    }
}
```

ザーへの表示（分数表示，小数表示，帯分数表示）が分離可能となる。

以上，要素が分数である行列を，JTable を利用していかに処理するかを述べた。特に，セルの内容とその表示が分離されていることにより柔軟なプログラミングが可能となっている。

5．プログラムの配布

前節までに，プログラムの概要を述べた。この節では，このプログラムをユーザーに実際利用してもらうために必要な残りの処理を説明する。

表１のクラス（これらのクラスの中に定義されているプライベート・クラスも含む）のソースファイル（拡張子は java）をコンパイルしバイトコード（拡張子は class）を作成し，これらを JAR（Java Archive）ファイルと呼ばれる１つのファイル（拡張子 jar）にまとめる。これを（アプリケーション版と呼ぶ）ユーザーに配布すれば，Java が実行できる環境のもとで，実行できる。

さて，この Java が実行できる環境というのが少しの問題を生む。本稿で作成した「行列の演算」プログラムの JAR ファイルは MatrixExercise.jar（142K バイト）で，かなり小さいファイルである。しかしながら，Java が実行できる環境にするために，Sun Microsystems 社の JRE（Java Runtime Environment）が必要である。JRE は以下の URL からダウンロードできる。

http://java.sun.com/j2se/1.3/ja/jre/

Windows の場合，この JRE をダウンロードしてインストールを行うと，なんと，21.9M バイトのハードディスクの容量を必要とする。142K バイトのプログラムを実行するために，21.9M バイトも必要なんて !!

JRE をインストール後，MatrixExercise.jar をダブルクリックすると（または，コマンドプロンプトから

＞java -jar MatrixExercise.jar [Enter]

と入力すると），図１のように「行列の演算」が起動される。

Java でプログラムしたために，わずか142K バイトのプログラムを実行する

ために，21.9Mバイトもの JRE を必要とした。これは，Java でプログラムした欠点であるが，次のような利点もある。すなわち，プログラムを変更せずに，そのままアプレット上で実行できる（アプレット版）のである。例えば，C++で同様のプログラムを作成したとしても，そのままアプレット上で利用することは出来ない。この節のはじめで述べた方法（アプリケーション版）では，MatrixExercise.jar を利用するユーザーが自分の計算機にこのファイルをダウンロードするなどして，保存しなければならなかった。従って，私がこのプログラムをバージョンアップしても，ユーザーがそれを新たにダウンロードしなければ，恩恵を受けることが出来ない。アップレットとしておけば，元のファイルはサーバー側にあり，それをバージョンアップするだけで，以後にこのアップレットにアクセスするユーザーはバージョンアップの恩恵を受けることになる。

アプレット上で起動するのは非常に簡単で，アプレット（MatrixExerciseApplet クラス）の init メソッドで，普通通り，MatrixFrame クラスを新しく作り，表示させるだけでよい（表3参照)。表3ではまず，System の LookAndFeel を設定し（これをしないと Windows 環境でも，Windows 風なウィンドウではなく Java 風のウィンドウになる)，画面の中央に「行列の演算」のメインウィンドウを表示している。

このアップレット（MatrixExerciseApplet.java）をコンパイルし，JAR ファイル（MatrixExerciseApplet.jar ；179K バイト）を作る。

HTML ファイルの Applet タグの部分は表4のようになる。「行列の演算」プログラムはこのアプレットにおいて，アプレット内ではなく独立したウィンドウとして表示されるので，width と height は1とした。

「行列の演算」プログラムは AWT （Abstract Window Toolkit）ではなく Swing と呼ばれる UI を利用しているので，現状では以上の準備のほかに，HTML ファイルの変換と Java Plug-in が必要である。HTML ファイルの変換は上記の HTML ファイルを JSDK （Java Software Development Kit）に添付されている HTML コンバータで新しい HTML ファイルに変換する。こ

表3 アプレット（MatrixExerciseApplet）

```
/**アプレットの初期化*/
public void init () {
  try {
    UIManager. setLookAndFeel (
        UIManager. getSystemLookAndFeelClassName ()) ;
    Dimension d = Toolkit. getDefaultToolkit (). getScreenSize () ;
    MatrixFrame matrixFrame1 = new MatrixFrame (false) ;
    matrixFrame1. setLocation ((d.width - matrixFrame1. getSize (). width) /
        2, (d.height - matrixFrame1. getSize (). height) / 2) ;
    matrixFrame1. setVisible (true) ;
  } catch (Exception ex) {
    ex. printStackTrace () ;
  }
}
```

表4 HTML ファイル（applet タグ）

```
< applet
  codebase = "."
  archive = "MatrixExerciseApplet.jar"
  code = "matrixexerciseapplet.MatrixExerciseApplet.class"
  name = "MatrixExercise"
  width = "1"
  height = "1"
  hspace = "0"
  vspace = "0"
  align = "top"
>
```

の変換された HTML ファイルと MatrixExerciseApplet.jar をサーバーの同じディレクトリに入れておく。この HTML ファイルをユーザーが Web ブラウザで表示すれば，JAR ファイルのダウンロードに暫く時間がかかるが，その後，

Webブラウザとは独立したウィンドウで「行列の演算」が起動される（図11参照)。「行列の演算」ウィンドウの下方のJava Applet Windowという表示がアップレットからこのウィンドウが起動されていることを示している。もし，Java Plug-inがユーザーのパソコン側にインストールされていなければ，Sunの Webサイトからダウンロードするかどうか尋ねられ，同意すると，Java Plug-inをダウンロードし，インストール後，「行列の演算」が起動される。同意しないと，「行列の演算」は表示されない。

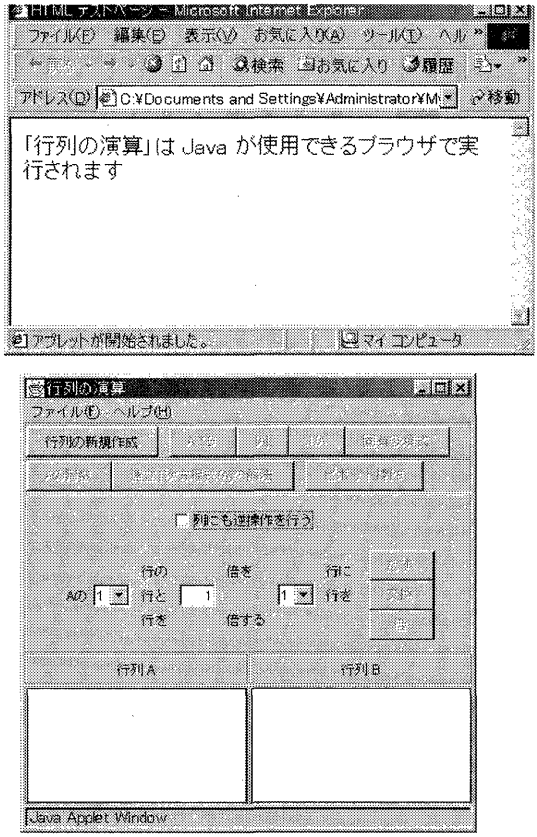


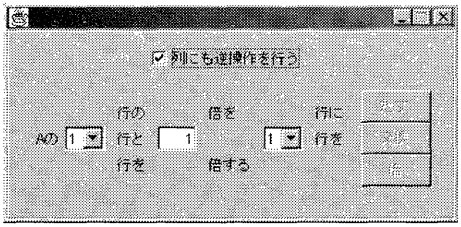
図11　アプレット版

Java Plug-in のインストールには JRE のインストールも含まれており，ユーザー側のパソコンのハードディスクをかなり専有することは，アプリケーション版での状況と同じである。

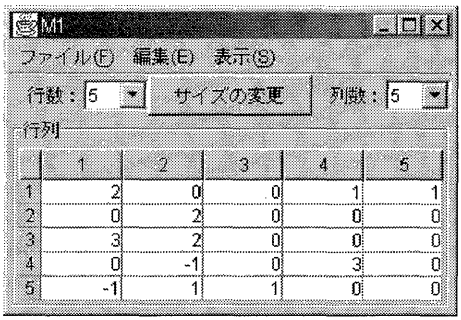
6．例

2．「行列の演算」プログラムの節で固有多項式を，ボタンを 1 回押すことによって求めたが，アルゴリズムを一歩一歩適用することによって，求めてみる。このアルゴリズムを適用して求めることができることは重要である。というのは，授業の受講生は，このアルゴリズムを利用して解くので，その途中経過をチェックできる必要があるからである。

さて，固有多項式を求めるアルゴリズムを適用してみる。利用するアルゴリズムに関しては［1］を参照。まず，次のようにメインウィンドウの「列にも逆操作を行う」をチェックする。

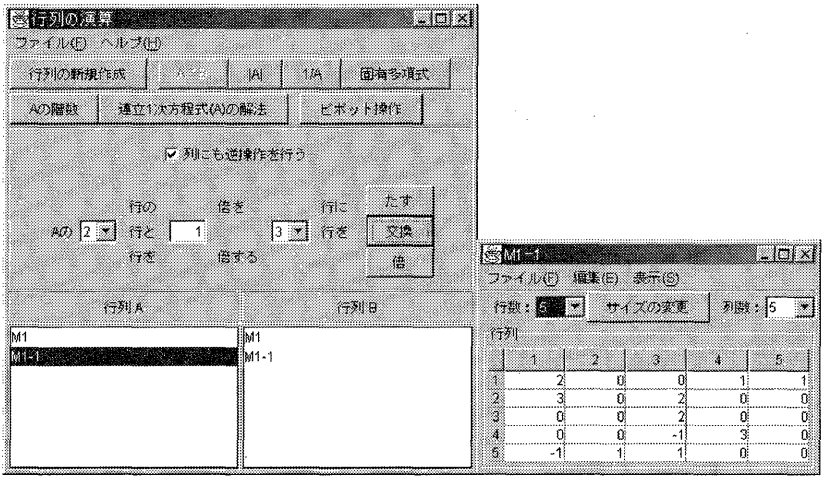


行列Aに行列が指定されれば，？行の？倍を？行に「たす」，？行と？行を「交換」する，？行を？「倍」するボタンを押すことが出来るようになる。



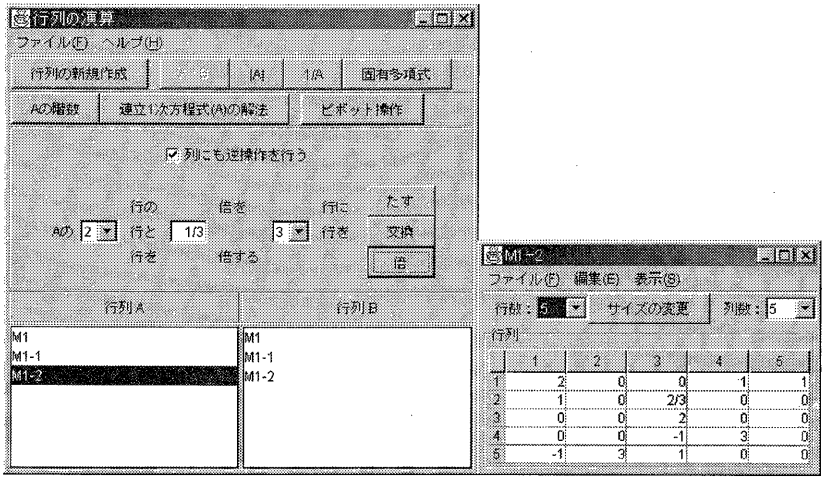
	1	2	3	4	5
1	2	0	0	1	1
2	0	2	0	0	0
3	3	2	0	0	0
4	0	-1	0	3	0
5	-1	1	1	0	0

- （必要ならば） M1を行列Aにして
- 2行と3行を交換

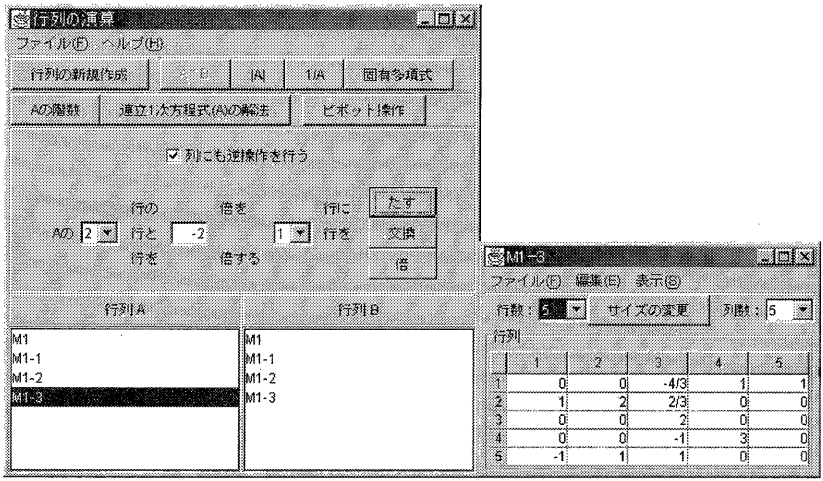


	1	2	3	4	5
1	2	0	0	1	1
2	3	0	2	0	0
3	0	0	2	0	0
4	0	0	-1	3	0
5	-1	1	1	0	0

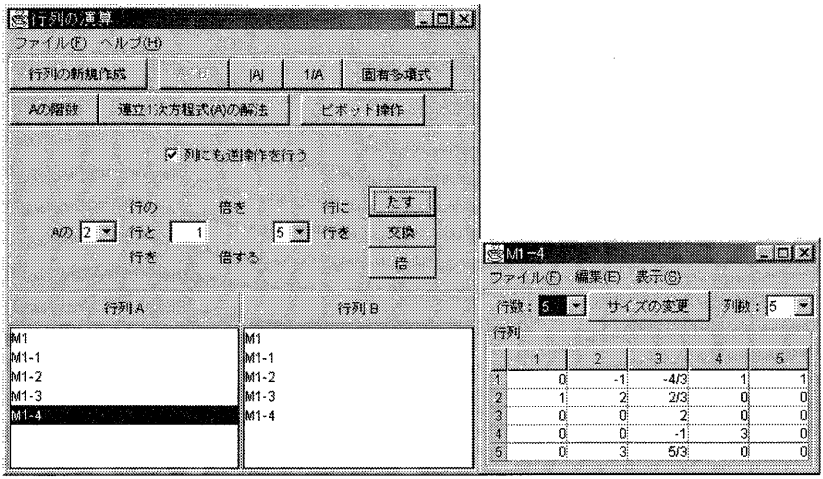
- （必要ならば） M1-1を行列Ａにして
- 2行を 1 / 3 倍する



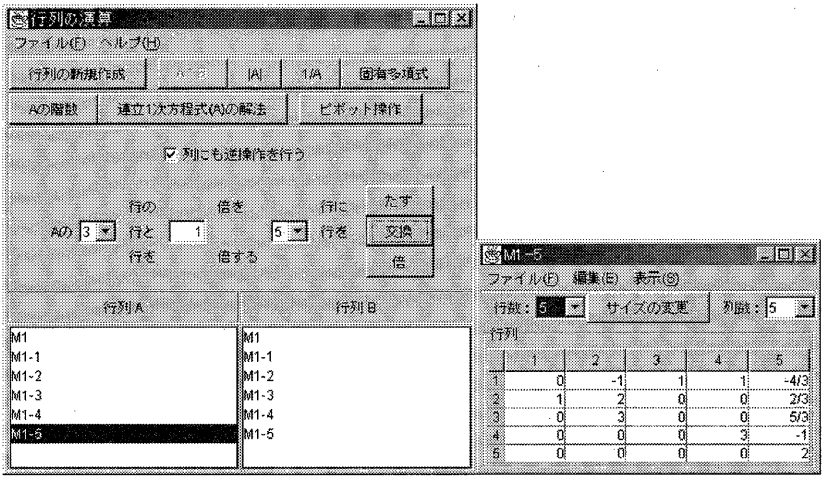
- （必要ならば） M1-2を行列Ａにして
- 2行の－2倍を1行にたす



- （必要ならば） M1-3を行列Aにして
- 2行の1倍を5行にたす

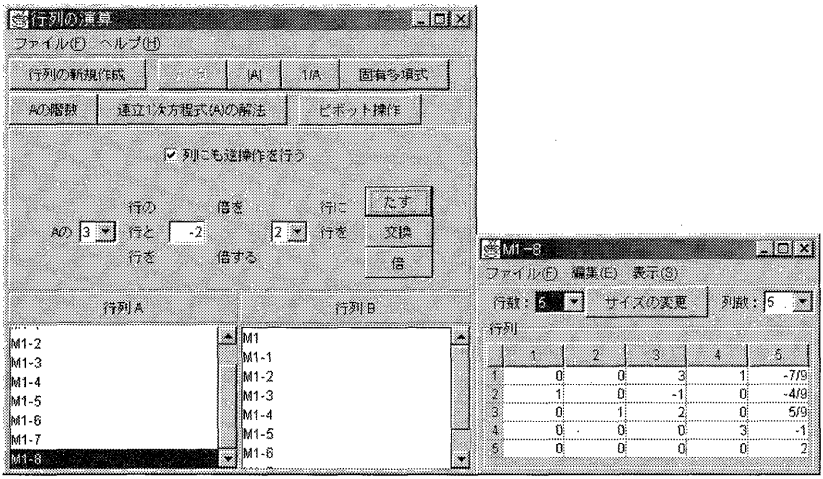


- （必要ならば） M1-4を行列Aにして
- 3行と5行を交換



- …（同様に行って）

- （必要ならば） M1-7を行列Aにして
- 3行の－2倍を2行にたす

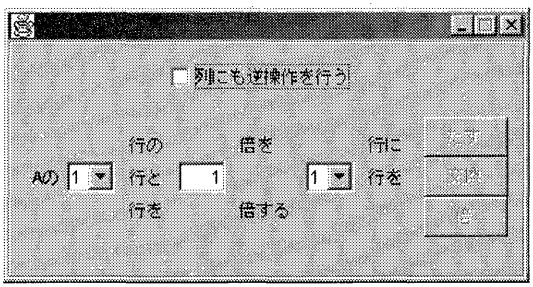


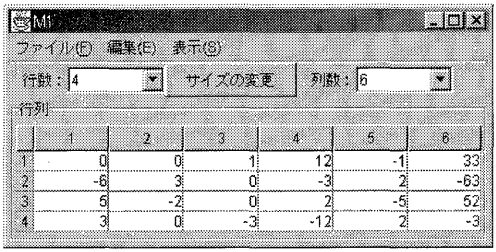
• 得られた行列 M1-8（図5と一致）が答えである。

以上において，計算経過がデスクトップ上に残っているので，自分のノートの解法と一致しているかチェックできる。また，途中に操作ミスをした場合でも，直前からやり直せばよい。

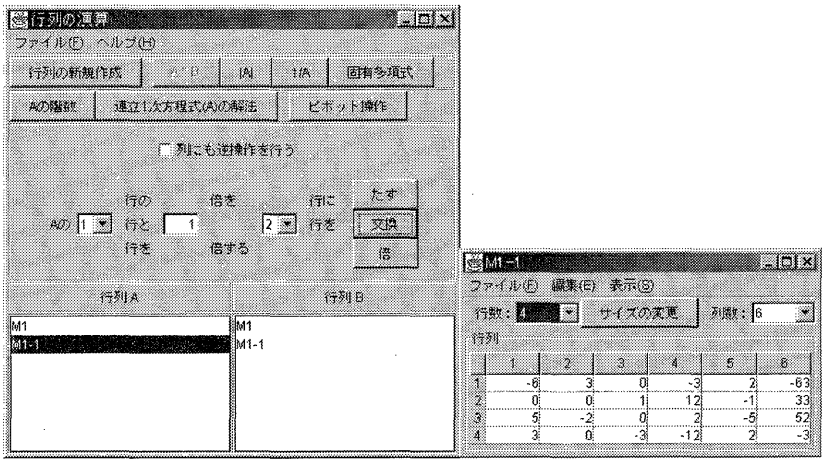
次に2.「行列の演算」プログラムの節でボタンを1回押すことによって求めた連立1次方程式を，アルゴリズムを一歩一歩適用することによって，求めてみる。

利用するアルゴリズムに関しては［1］を参照。まず，必要ならば次のようにメインウィンドウの「列にも逆操作を行う」をチェックをはずす。

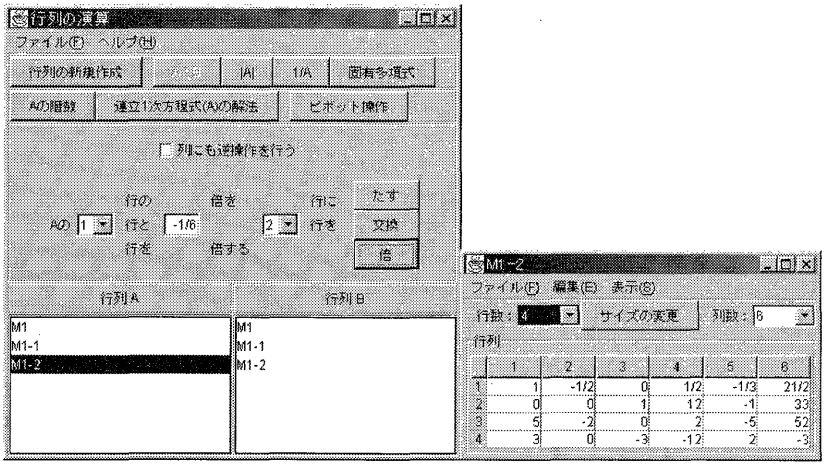




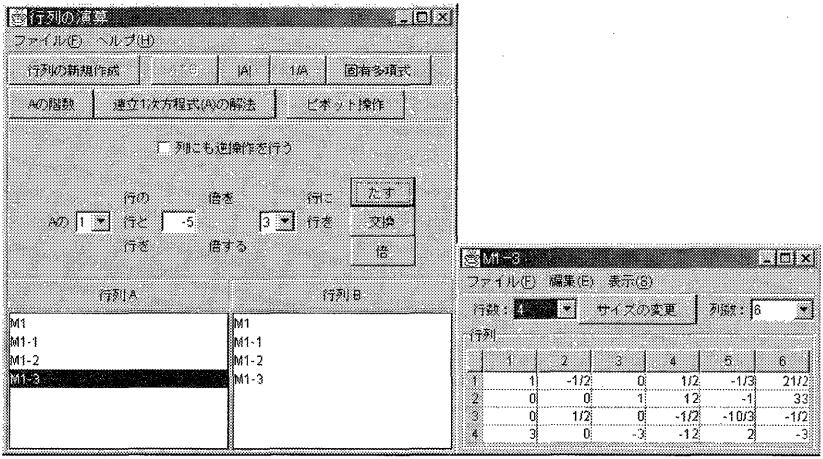
- （必要ならば） M1を行列Aにして
- 1行と2行を交換



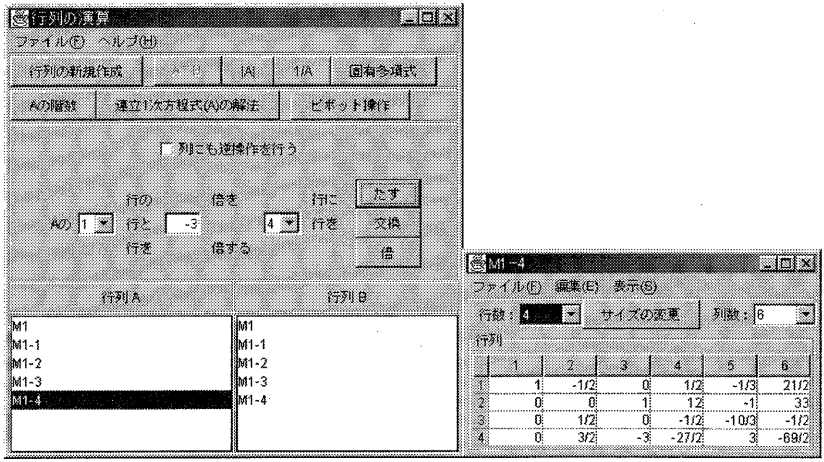
- （必要ならば） M1-1を行列Ａにして
- 1行を－1/6倍する



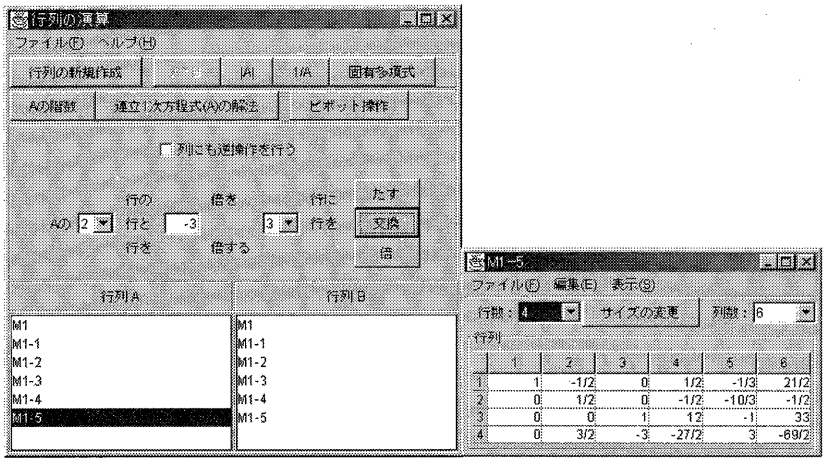
- （必要ならば） M1-2を行列Ａにして
- 1行の－5倍を3行にたす



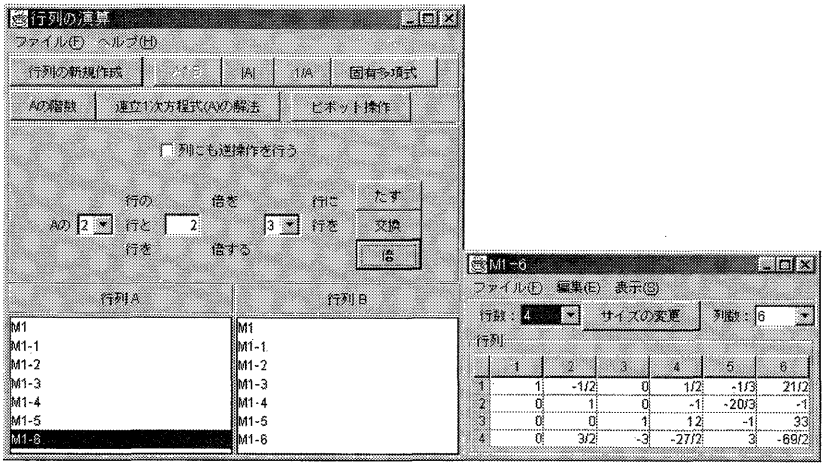
- （必要ならば） M1-3を行列Aにして
- 1行の－3倍を4行にたす



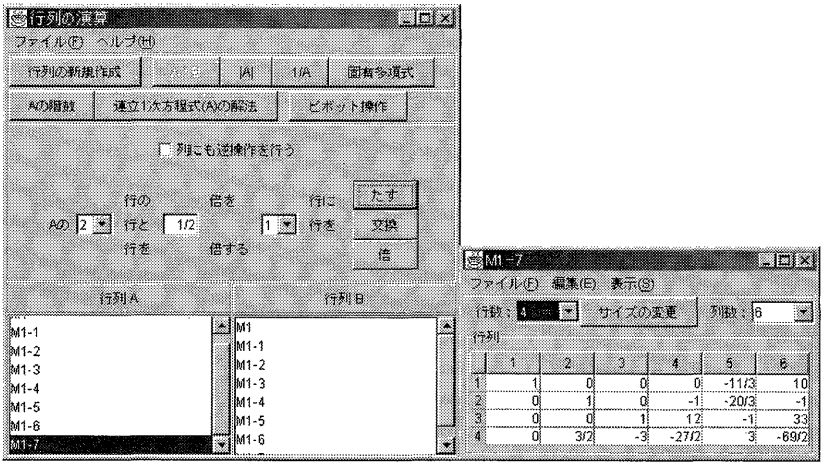
- （必要ならば） M1-4を行列Aにして
- 2行と3行を交換



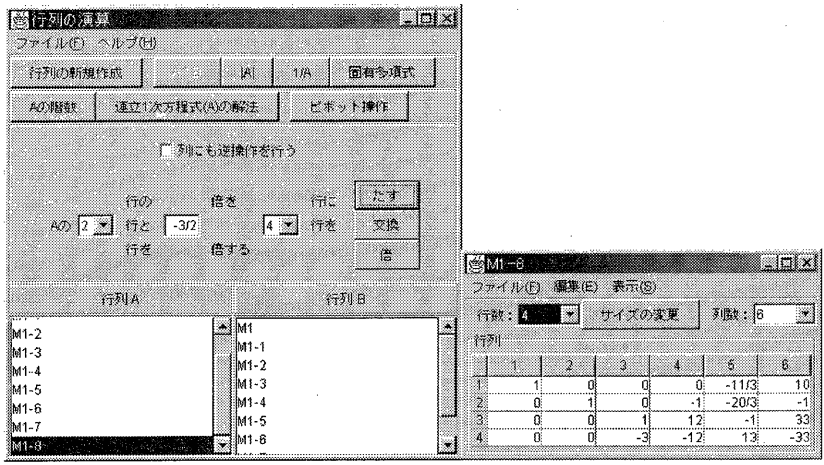
- （必要ならば） M1-5を行列Ａにして
- ２行を２倍する



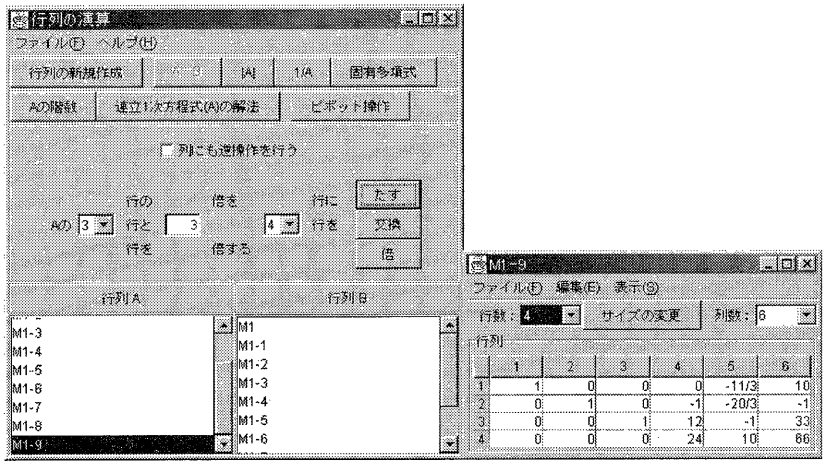
- （必要ならば） M1-6を行列Ａにして
- ２行の１／２倍を１行にたす



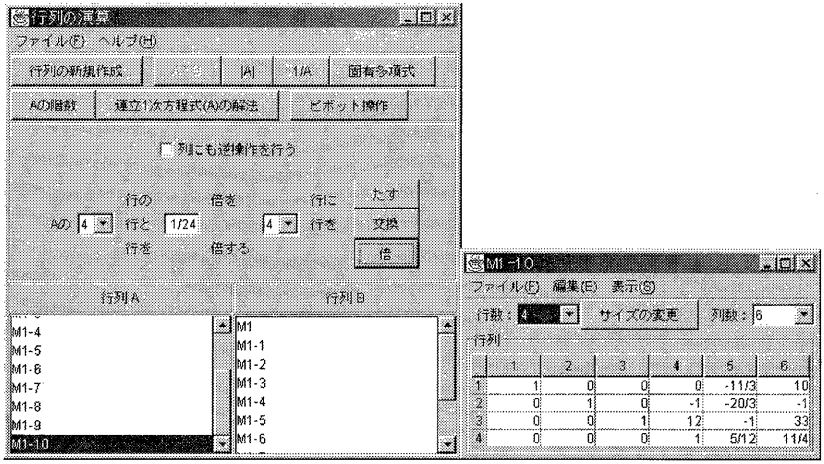
- （必要ならば） M1-7を行列Aにして
- 2行の－3/2倍を4行にたす



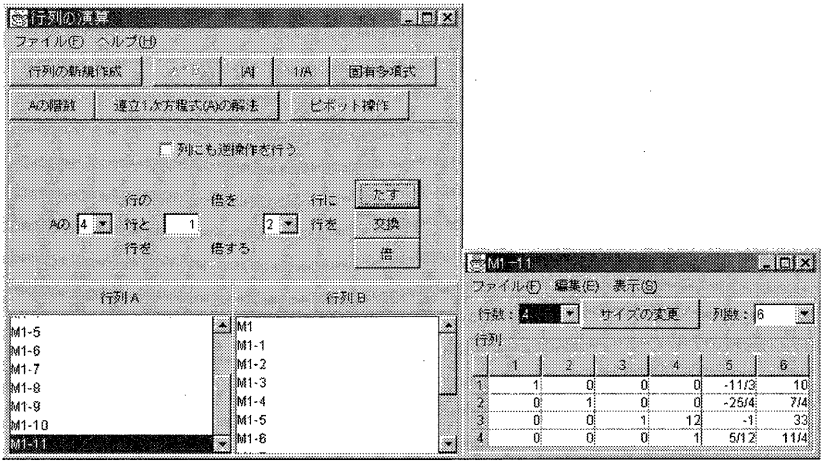
- （必要ならば） M1-8を行列Aにして
- 3行の3倍を4行にたす



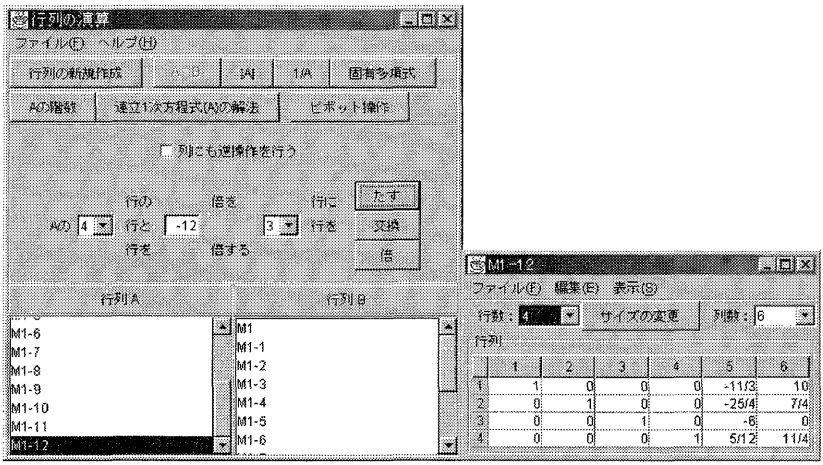
- （必要ならば） M1-9を行列Aにして
- 4 行を 1 /24倍する



- （必要ならば） M1-10を行列Aにして
- 4 行の 1 倍を 2 行にたす



- （必要ならば） M1-11を行列Aにして
- 4行の－12倍を3行にたす



- 図7の解が得られた。

7．おわりに

本稿では授業を補助するプログラム「行列の演算」を紹介した。Java言語で作成した利点をまとめておく。作成したMatrixExercise.jarはJavaが実行できる環境であれば，Windows以外のOS上でも作動する。また，「行列の演算」のメインウィンドウであるMatrixFrameクラスをアプレットから直接呼び出すことが出来る。

最後にどれくらい安価に作成できたかについて述べる。私はJBuilder Professional版を用いたため，これを購入する費用が必要であった。しかし，これ以外は必要ではなかった。このような製品を利用する利点は，部品をマウスでドラッグアンドドロップすることでUIを簡単に作成できることである。このような製品がなくても，JSDKをSunのWebサイトから無償でダウンロー

ドしてくれれば，それとテキストエディタだけで，本稿で紹介したプログラムは作成できる。私の驚きは，テーブルの豊富な機能とその簡単な利用法であった。参考文献にある本とヘルプを参考にするだけで，本稿で紹介したプログラムを作成することが出来た。私は以前 VisualBasic と C ++ Builder を利用したことがあるが，そこに標準として添付されているテーブルでは満足な処理が出来ず，結局，サードパーティから発売されている部品を購入しないと自分の目的を遂げることができない状況であった。現在のような，安価で豊富な機能を利用できる Java の状況が長く続くことを祈っている次第である。

参考文献

[1] 沼田・行方他「線形数学」富士書院，1991。
[2] ミッシェル・マニング著／玉川竜司訳「JBuilder で学ぶ Java」プレンティスホール，1998。
[3] 掌田津耶乃「JBuilder ではじめる Java プログラミング入門」秀和システム，2001。
[4] Mary Campione, Kathy Walrath, Alison Huml : The Java Tutorial, Third Edition-A Short Course on the Basics. Addison Wesley, 2001.
[5] Kathy Walrath, Mary Campione : The JFC Swing Tutorial-A Guide to Constructing GUIs. Addison Wesley, 2000.
[6] Campione, Walrath, Huml, Tutorial Team : The Java Tutorial Continued-The Rest of the JDK. Addison Wesley, 1998.