計算機代数の講義と試験における計算機の利用について[1]

神戸大学・人間発達環境学研究科 長坂 耕作
Kosaku Nagasaka
Graduate School of Human Development and Environment,
Kobe University

1　はじめに

本稿では，神戸大学国際人間科学部環境共生学科の 3,4 年生配当の学科展開科目「計算代数 B」において，講義と定期試験の双方で積極的にパソコンを活用した事例について報告したい（本科目のシラバスは表 1 を参考されたい）。なお，誤解のないようにシラバスと重複するものの言及しておくが，本授業科目は「理論と算法」に関する科目であり，決して，計算機を用いて問題を解くという科目ではないことに留意頂きたい。

1.1　計算機の利用を前提とした経緯

計算代数 B の令和 4 年度の授業形式は，反転授業形式を採用しており，Moodle 上で配布される授業資料（PDF 資料と動画資料）による教室外の講義部分（事前学修）と，教室内の演習部分（質疑応答や補足説明，学生間の意見交換等）により構成している。本年度の履修者数が少なかったこともあり，定期試験までも計算機の利用を前提とした授業展開を試みたものである。なお，本授業科目は隔年開講であり，前回の開講時は COVID-19 の感染拡大初期で遠隔授業（リアルタイム）の反転授業として実施しているが，計算機の利用は前提としていなかった（ただし，演習時の利用は禁止していない）。

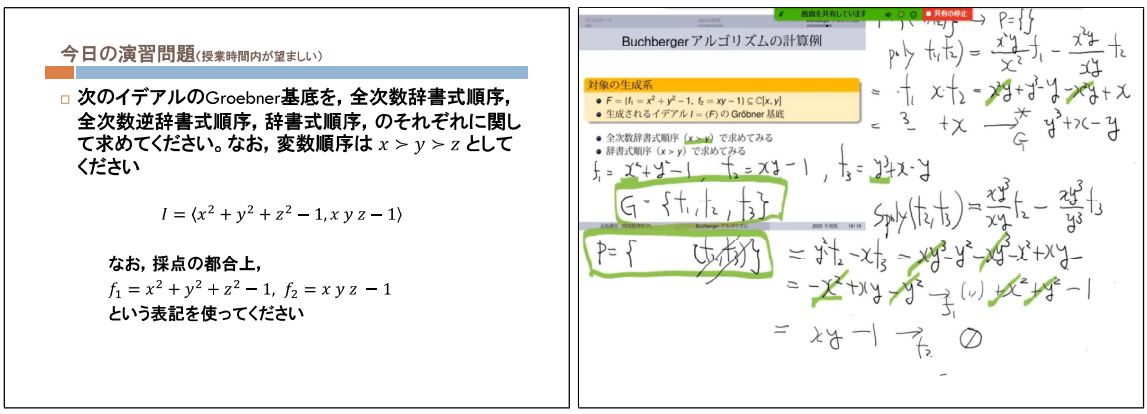

図 1: Gröbner 基底を計算する Buchberger アルゴリズムに関する授業資料

[1]This work was partially supported by JSPS KAKENHI Grant Number 21H00921.

開講年次	3,4 年	開講区分	前期	曜日時限等	木 3	単位数	2.0
授業のテーマ	不定元が複数の代数方程式（多変数の代数方程式）を解くことに必要な理論や算法（アルゴリズム）について学ぶ。						
授業の概要と計画	第 1 回: 線形方程式と代数方程式 第 2 回: 線形方程式と掃きだし法 第 3 回: 多変数多項式の除算と項順序 第 4 回: LU 分解 第 5 回: Gröbner 基底とその性質 第 6 回: LU 分解と枢軸選択 第 7 回: Buchberger アルゴリズム 第 8 回: 線形方程式と最小二乗法 第 9 回: 極小 Gröbner 基底と簡約 Gröbner 基底 第 10 回: QR 分解 第 11 回: イデアルと Affine 多様体 第 12 回: QR 分解と枢軸選択 第 13 回: 代数方程式と Gröbner 基底 第 14 回: 線形方程式と高速算法 第 15 回: Buchberger アルゴリズムの高速化 ※ 線形方程式の数値解法と非線形方程式の厳密解法を隔週						
成績評価方法	各回の内容を確認するレポート課題（20 %）と最終試験（80 %）を行う。ただし，授業への積極度（質疑や LMS の活動ログ等）が著しく低い場合は最大 5 割の減点を行う。						

表 1: 計算代数 B のシラバス（抜粋）

授業では幾つかのアルゴリズムとその証明を与えるが，アルゴリズムの動作や表れる語句の理解を深めてもらうために，実際にアルゴリズムを用いて計算を行う演習も設定している。例えば図 1 は，多項式環のイデアルの Gröbner 基底（非線形の代数方程式の解法や数式の簡単化などに用いられる）を計算する演習問題（図の左）と，動画資料で計算例を説明している場面の切り出し（図の右）である。

図 1 の演習問題では，手計算で可能な規模の問題となっており，かつ，アルゴリズム自体の働きを理解するためにも手計算で行って欲しいと考えている。実際に，動画資料の切り出した場面では，Gröbner 基底計算で重要な概念である項順序に関する理解が求められており，それを演習時に確認するためにも省略することは難しい。

一方，授業の終盤では，Gröbner 基底を用いて他の計算を行う場面が増える。図 2 は動画資料を切り出した画像であるが，有理関数体上で Gröbner 基底を計算（ラジカルメンバーシップ問題と呼ばれる問題を解くことに相当）することで，アポロニウスの円の定理を証明している場面となる。この段階で行う演習問題では，演習の対象となる初等幾何の性質が簡易であっても，Gröbner 基底に帰着した段階において，手計算で行うことが厳しくなることが多い。実際，図 2 の右側に見られるように，有理関数体上の多項式環での計算が必要となるため，非常に煩雑となってしまう。この煩雑となる計算部分

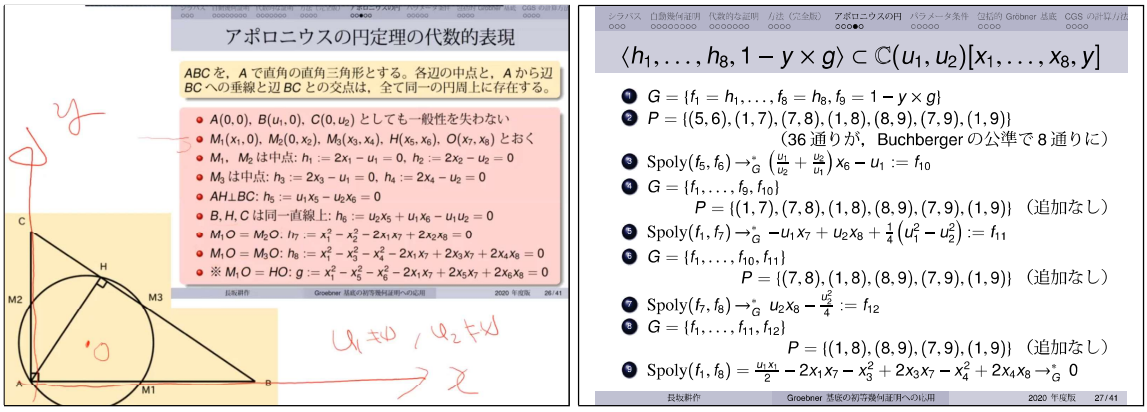

図 2: 有理関数体上で Gröbner 基底を計算する初等幾何の証明に関する授業資料

は当該授業回の主目的ではないため，手計算に替え，計算機を用いて演習を行うことが望ましいと考えられる。

以上のことから，理論と算法に関する授業科目ではあるが，計算機の利用を前提とした講義と定期試験を令和 4 年度に試みることとした。なお，本科目は COVID-19 前に開講はされておらず，対面での開講は令和 4 年度が初となる（国際人間科学部設置後初回の開講時が COVID-19 の初期であったため）。加えて，神戸大学では平成 31 年度入学生からパソコン必携化を行っている関係で，授業内容が情報科学等に関係なかったとしても計算機の利用を前提とすることは問題とならない。

1.2 どのような計算機の利用を前提とするか

授業では，Wolfram Alpha または Python の SymPy を活用するように指示したが，実際に利用されていたのは Wolfram Alpha だけであった（一応，履修者は 2 年生の授業時に，Python の SymPy に触れており，大学入試の問題を解く演習も行っている）。なお，Wolfram Alpha に関して演習時にあらかじめ説明した内容は，図 1 の演習時であれば，次の 3 点となる。

- 式の展開の仕方
- `GroebnerBasis` の命令の使い方（Gröbner 基底を計算する）
- `PolynomialReduce` の命令の使い方と出力の見方（単項簡約を行う）

2 計算機の利用を前提とした定期試験

実際に使用した定期試験が図 3 である。授業目的である不定元が複数の代数方程式を解くことに必要な理論や算法についての理解を確認するため，計算機の利用を前提とした問題となっている。定期試験結果の詳細については触れないが，解答状況を見る限りにおいては，本来の授業目的を達成することに関して，計算機の利用を前提としたことによる悪影響はないように見受けられた。念のため，問題の解説を行っておく。

問 1 では，大学初年次級の線形代数の問題（係数行列と拡大係数行列の階数から解を持つかを判定）から始まり，過剰決定系を QR 分解で解く方法を理解しているか，解を唯一つ持つ線型方程式を LU 分解で解く方法を理解しているか，そして，これらの解の性質の違いを理解しているかを確認している。Wolfram Alpha の利用を前提としているため，Mathematica のコマンドで言えば `RowReduce`, `QRDecomposition`, `LinearSolve`, `LUDecomposition` などの利用を想定している（これらの命令の利用方法は諸注意として問題用紙に掲載した）。

問 2 では，Gröbner 基底を計算する Buchberger アルゴリズムと，最小多項式を用いて代数方程式の解を求める方法の理解を確認している。この問題では，Mathematica のコマンドで言えば `LinearSolve`, `PolynomialReduce` などの利用を想定している。

問 1 ある性質を表す $\vec{x} \in \mathbb{R}^3$ の値を求めるための実験を行い，その結果を小数第 2 位まで計測したところ，$\vec{x}$ に関する次の線形方程式を得た。この線形方程式を活用して $\vec{x}$ の値を求めることに関して以下の問に答えよ。

$$\begin{pmatrix} -0.31 & -0.38 & -0.87 \\ 0.31 & 0.29 & 0.58 \\ -0.19 & 0.70 & 0.60 \\ 0.98 & -0.82 & 0.30 \end{pmatrix} \vec{x} = \begin{pmatrix} 0.43 \\ -0.29 \\ -0.26 \\ -0.20 \end{pmatrix}$$

（a）この線形方程式は解を持つか確認せよ。

（b）QR 分解を用いて，上記の線形方程式をある程度満たす $\vec{x}$ を求めよ。

（c）条件を緩和するため 4 行目の制約を無視し，1 行目から 3 行目までの制約を満たす $\vec{x}$ を求めることにした。実際に PLU 分解を求め，三角行列を係数行列とする方程式を繰り返し解いて，この条件の $\vec{x}$ を求めよ。

（d）上記 (a),(b),(c) で解の状況は同一ではないが，それは何故か。それぞれの解の性質を述べつつ説明せよ。

問 2 次の代数方程式の解を Gröbner 基底と最小多項式を使用して求めたい。

$$\begin{cases} -x^2z - x^2 + 2xyz + 2xy + 3xz + 3x - y^2 - 6yz - 3y - 2 &= 0 \\ -x^2z + 2xyz + 3xz - 6yz + z - 3 &= 0 \\ -x^2 + 2xy + 3x + y^2z - y^2 - 3yz - 3y - z^2 + 5z - 2 &= 0 \end{cases}$$

上から順に左辺の多項式を f_1, f_2, f_3 とおき，イデアル $\langle f_1, f_2, f_3 \rangle$ の辞書式順序（$x \succ y \succ z$）での Gröbner 基底を Buchberger アルゴリズムで途中まで計算し，次のところまでは計算が終わっている。続きの計算を行い，実際に解を求めよ。

- 追加された多項式: $\{f_4 = y^2z - 3yz - z^2 + 6z - 3,\ f_5 = z^2 - 3z,\ f_6 = z - 3,\ f_7 = y^2 - 3y + 2\}$
- S 多項式の計算の終わっているペア: (f_1, f_2), (f_1, f_3), (f_1, f_4), (f_1, f_5), (f_1, f_6), (f_2, f_3), (f_2, f_4), (f_2, f_5), (f_2, f_6), (f_3, f_4), (f_3, f_5), (f_3, f_6), (f_4, f_5), (f_4, f_6), (f_5, f_6)

図 3: 実際に使用した定期試験の問題用紙（諸注意等を除く問題部分の抜粋）

3 WolframAlpha にするか Colab にするか問題

前章までに紹介した取り組みは機能しているように見受けられるため，次年度以降も積極的に継続・発展させたいと考えている。しかしながら，ここで課題として挙げられるのが，学生に使用を促す環境である。学生側及び大学側の双方において無償かつ容易に利用可能な環境を前提に考えると，実質的には二者選択となる。すなわち，Wolfram Alpha にするか，Google Colaboratory にするか，である。なお，Python の実行環境としては後者以外の選択肢も存在するが，利用の容易さという面で除外した（神戸大学でも「Google Workspace for Education」のアカウントが発行されていることも考慮）。

次年度以降の利用を想定してメリット・デメリットをまとめると次のようになる。

Wolfram Alpha のメリット（+）・デメリット（−）

+ 適当に入力しても補正して解釈してくれる
+ ログインとか面倒なことはほとんどない
− 入力できる文字数に制限がある
− プログラミング的な操作は難しい（コピペで対応）
− Gröbner 基底関係だと項順序が辞書式に固定

Google Colaboratory のメリット（+）・デメリット（−）

+ Python や SymPy の機能をフルに活用できる
− Python の文法に沿って使わないといけない
− 少し間違いも許容されず，空気を読んでくれない
− 掛け算を省略できないし，指数は「**」

以上のメリット・デメリットを考慮すると，Wolfram Alpha に不足している機能は，Wolfram One などのクラウドサービスを利用することで解決するが，Wolfram Cloud の無償アカウントは授業利用を禁止しているようなので，無償で容易という条件からは外れますし，Python の文法ではないものの Mathematica の文法を覚える必要もある。一方で，Google Colaboratory に不足している機能は，Python やベースとなっている Jupyter notebook に起因している。すなわち，Python や Jupyter notebook の作法に則ることで，不足している部分を補える可能性が存在する。以下はこの可能性を追求した結果である。

3.1 簡易文法修正マジックコマンドの開発

Jupyter notebook はインターフェイスであるため，Python のカーネルで行われる計算処理以外にも，ノートブック上の挙動を操作することが可能となっている（マジックコマンド）。また，LaTeX などのリアルタイムレンダリングによる高品質な数式表示も対応しており，SymPy を用いれば，数式の LaTeX への変換も容易であるし，Python の標準ライブラリに含まれる ast モジュールを用いれば，Python の式を構文解析した結果を用いることも可能となる。これらの機能を活用して，ある程度空気を読んだ解釈を行い，適切な Python の命令を実行するマジックコマンドを開発中である。Google Colaboratory にアップロードして使用する前提で用意したノートブックが次の URL からダウンロード可能である。

- `https://wwwmain.h.kobe-u.ac.jp/~nagasaka/research/rims22edu/簡易文法修正マジックコマンド.ipynb`

図 4: 簡易文法修正マジックコマンドの実行例

このノートブックでは，図 4 の冒頭の説明にあるように，コードを非表示にしているセルを最初に一度だけ実行する必要がある。このセルでは，簡易的に文法を修正し，修正後の命令や数学的な慣用表記などを表示するマジックコマンド（`%mm`）を定義している。図の後半にある利用例のように，簡易的な文法修正を行うマジックコマンド「`%mm`」に引き続き，Python の SymPy を利用した計算式を入力することで，仮に文法が間違っていても修正を行った上で計算をする。図にあるように，修正後の命令，修正後の命令を数学的な表記にしたもの，計算結果をコピー＆ペーストしやすく表示したもの，実際の計算結果，が順に表示されるようになっている。

この図の例では，1) 「`expand`」という命令を誤って「`expan`」と書いていること，2) 関数の呼び出しであるが丸括弧を省略してしまっていること，3) 「`2*a`」と書くべきところで乗算記号を省略してしまっていること，4) 「`x*y`」と乗算記号を挿入すべきを「`xy`」と連続して書いていること，5) 式全体の二乗を表すのに「`**`」ではなく「`^`」を使用していること，の 5 つの文法上の間違いがあり，修正されている。

上記ノートブックに掲載している他の例も簡単に紹介しておく。図 5 では，指数を表す際の「`^`」の誤用の修正をしている。図 6 では，空白を乗算記号のようにしようする誤用や，添字への変換を行っている。図 7 の 2 つの例では，SymPy の行列型への変換や，行列やベクトルの計算での乗算記号の省略を修正している。図 8 では，SymPy の関数名の修正や，SymPy の行列型への変換を行っている。図 9 では，丸括弧を省略した関数呼び出しを修正している。また，この例ではコピー＆ペーストするための出力の必要性が確認しやすい。図 10 の 2 つの例では，分数の取り扱いを修正している（Python でそのまま「`/`」を使用すると小数になってしまう）。

図 5: 簡易文法修正マジックコマンドの実行例（指数の扱い）

図 6: 簡易文法修正マジックコマンドの実行例（空白や添字の扱い）

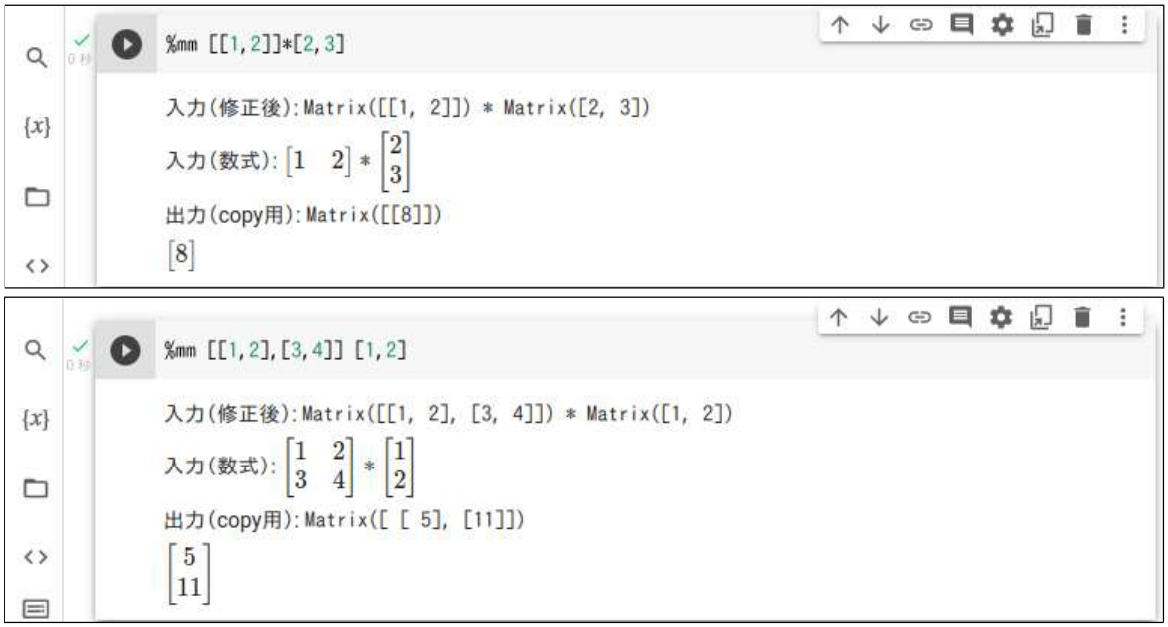

図 7: 簡易文法修正マジックコマンドの実行例（行列の積やベクトルの解釈）

4　まとめ

この報告では，計算機代数の講義と試験における計算機の利用を前提とした授業展開について紹介し，その上で課題となった「無償かつ容易を前提とするならば，どのシステムを利用すべきか」の解決策の一案として，開発中の簡易文法修正マジックコマンドについて紹介した。ただ，会場からの意見にもあったように，Wolfram Alpha や簡易文法修正マジックコマンドのような，本来の言語仕様からは不適切と考えられる入力を受け入れてしまうことは，教育的であるかの疑問が残る。今後の課題として検討したい。

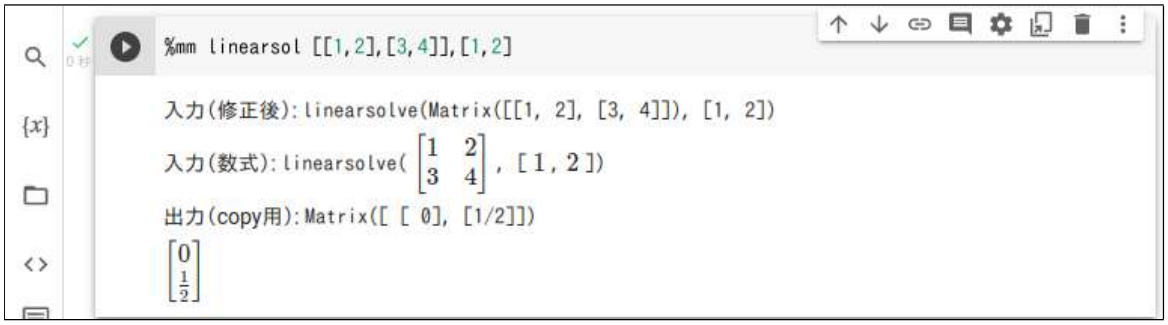

図 8: 簡易文法修正マジックコマンドの実行例（線形方程式の求解）

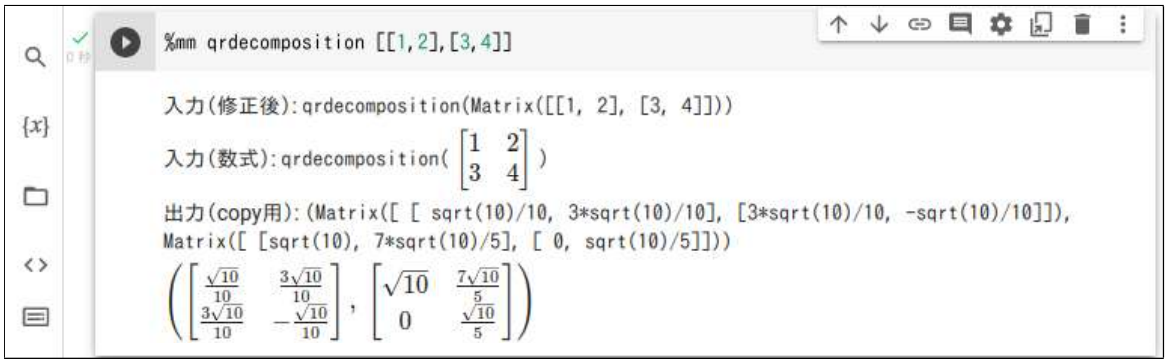

図 9: 簡易文法修正マジックコマンドの実行例（QR 分解）

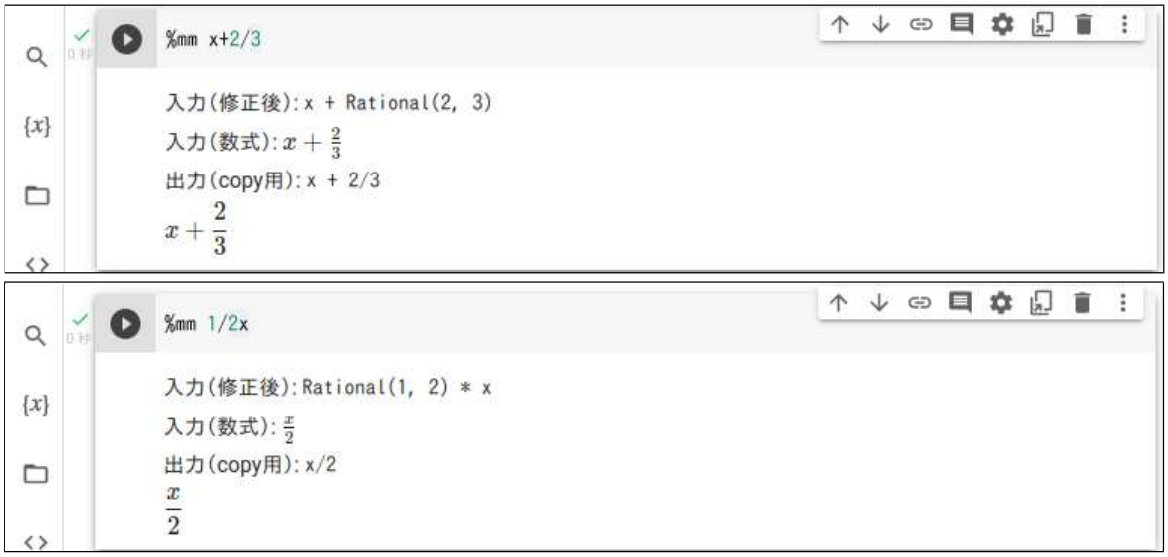

図 10: 簡易文法修正マジックコマンドの実行例（分数の取り扱い）