P-19

機械学習を用いたグレブナー基底の計算
Computing Gröbner Bases Using Machine Learning

○石原侑樹 [1]
*Yuki Ishihara[1]

Abstract: Gröbner bases are very useful tool to solve systems of polynomial equations. However, it is known that its computation tends to be very time-consuming. The purpose of this article is to explain a method by machine learning and effective techniques for generating data set for the learning. The results in this article are based on the paper [2].

1. はじめに

近年，生成 AI を用いた文章や画像の生成が活発に行われている．ChatGPT を含む生成 AI の多くは “Transformer” [3] と呼ばれる深層学習モデルを用いている．最近では，数学の計算を Transformer で行う研究も盛んに行われてきており，例えば，行列の計算，積分の計算，常微分方程式の求解などの学習が挙げられる．本稿では，グレブナー基底の計算を Transformer で学習した研究について紹介する．研究成果は著者の共著論文 [2] を引用している．

2. 記号計算と機械学習

素因数分解，微分積分，連立方程式の求解，行列の計算など，高校や大学で習う数学の計算の多くは，Mathematica などの数式処理ソフトウェア上で計算することができる．それらのアルゴリズムは主に「記号計算」と呼ばれる誤差を生まない計算をベースとしている．例えば，数値計算においては,「2 の正の平方根」は 1.41421356 といった有限小数で扱うことが多いが，記号計算においては $\sqrt{2}$ のように「2 乗したら 2 になる正の数」として記号的に扱う．

記号計算は，数学的な計算を厳密に扱えるメリットがある一方，数値計算に比べて計算コストが高いデメリットがある．その欠点を解消するために，近年特に注目されている方法の 1 つが機械学習である．通常の場合，数学的な計算をコンピュータで行う際には，事前に人間がアルゴリズムを考案し，その正当性や停止性を数学的に証明した上で，計算を実行する．例えば，素因数分解において，入力（自然数 N）に対し小さい整数 p から順に p が N を割り切るか判定し，割り切れた場合には N/p に対し同様の処理を行っていく（素朴なアルゴリズムの場合）．

一方，機械学習による典型的な方法では，入力（自然数 N）と出力（N の素因数分解）の組のデータセットを基に学習し，その学習結果から新しく与えられた N の素因数分解を計算する．すなわち，人間が 1 からアルゴリズムを考えるのではなく，学習によって入力（問題）と出力（答え）の間の規則性を見つけ，そこから未知の入力に対しても計算を行う．機械学習による計算のメリットとしては，計算手法が確立していない計算問題にも適用できることや，従来の手法よりも高速に計算できる可能性があることが挙げられる．その反面，デメリットとして，出力の正当性（精度）の評価が必要なことや，学習のためのデータセットの準備を要することが挙げられる．本稿では，特に後者のデータセットの生成の問題について議論する．

3. データセット生成の問題点と解決方法

先述したように，機械学習で学習する際には通常，事前に膨大な量のデータセットの準備が必要になる．しかし，既に実装されている記号計算のプログラム（素因数分解，微分積分，グレブナー基底…）を用いて，データを生成する場合には，多大な時間を要する傾向がある．例えば，1 個の具体例を計算するのに 1 分かかるプログラムがあるとして，100 万個のデータセットを作るには 100 万分（694 日）かかることになる．そのため実用的な時間内で学習が終わらないという問題が生じる．このように，時間がかかる処理の高速化を機械学習で行うための，データセットの生成に多大な計算時間がかかるという「データセット生成のパラドックス」と言えるような状況が起こりうる．

このパラドックスを回避する方法の 1 つとして，データ生成の方法を「逆順」にするという手法が考えられる．つまり，先ほどのパラドックスにおいては,「入力」から「出力」を作っていたため多くの計算時間を要したが，逆に「出力」から「入力」を作ってしまえば計算時間を大幅に短縮できる可能性がある．例えば，素因数分解においては，入力（整数 N）からその出力（素因数分解 $N = p_1^{e_1} \cdots p_r^{e_r}$）を計算することは時間がかかるが，出力（素数の積 $p_1^{e_1} \cdots p_r^{e_r}$）から入力（整数 $N = p_1^{e_1} \cdots p_r^{e_r}$）ははるかに素早く計算できる（この素因数分解の計算の非対称性は RSA 暗号などにも用いられている）．同じように，特定のクラスにおける積分の計算において，入力（関数 $f(x)$）から積分（$\int f(x)\,dx$）を求めることは難しいことが多いが，出力（$\int f(x)\,dx$）から入力（関数 $f(x)$）を得るには，$\int f(x)\,dx$ を微分すればよいので，比較的容易なことが多い．このように，計算問題の「逆順」を計算することによる「逆順生

1: : 日大理工・教員・数学

成（backward generation）」と呼ばれる手法は，数学の計算を機械学習する場面において効果的なテクニックになると考えられる．

4. グレブナー基底の学習

節 3 で述べたデータセット生成の問題は「グレブナー基底」にも当てはまる問題である．グレブナー基底は多項式イデアルの特別な生成系であり，連立方程式の求解やイデアルの不変量の計算を始め，暗号理論，制御理論など多岐に渡り応用されている概念である．しかし，グレブナー基底の計算は記号計算をベースとしており，最悪計算量が 2 重指数的であるという欠点を持つ．そこで，機械学習による高速化を目指すことが，論文 [2] における目的である．

以降，k を体，$k[x_1,\dots,x_n]$ を k 上の n 変数多項式環とする．また，$k[x_1,\dots,x_n]$ の単項式全体を $\mathcal{T}=\{x^{\alpha_1}\cdots x^{\alpha_n} \mid \alpha_1,\dots,\alpha_n$ は非負整数 $\}$ とし，$\mathcal{T}$ 内の単項式順序 $\prec$ を考える．すなわち，$\prec$ は整列順序であり，単項式の積に関して順序を保存する．また，$f\in k[x_1,\dots,x_n]$ に対し，$\mathrm{LT}_\prec(f)$ で $\prec$ に関する f の先頭項を表す．また集合 $S\subset k[x_1,\dots,x_n]$ に対し，$\langle \mathrm{LT}_\prec(S)\rangle=\langle \mathrm{LT}(f)\mid f\in S\setminus\{0\}\rangle$ とする．ここで，$\langle X\rangle$ は X から生成されるイデアルを表す．

定義 1 (グレブナー基底, [1]) I を $k[x_1,\dots,x_n]$ のイデアルとする．I の有限部分集合 G に対し，$\langle \mathrm{LT}(G)\rangle=\langle \mathrm{LT}(I)\rangle$ が成り立つならば G は $\prec$ に関する I のグレブナー基底と呼ばれる．

例 2 $I=\langle 3x+4y-3, 9xy+13y^2\rangle\subset\mathbb{Q}[x,y]$ に対し，$\prec$ を辞書式順序 $y\prec x$ とすると，$G=\{3x+4y-3, y^2+9y\}$ は $\prec$ に関する I のグレブナー基底である．ここで，$I=\langle G\rangle$ が成り立つため，G は I の生成系を連立方程式が解きやすい形に変形したものになっている．実際，G は 1 変数多項式 y^2+9y を含むため，$y^2+9y=0$ を解くと，$y=0,-9$ が成り立ち，$3x+4y-3=0$ に代入して x を求めると，$(x,y)=(1,0),(13,-9)$ が連立方程式 $3x+4y-3=9xy+13y^2=0$ の解であることが分かる．

イデアル $I=\langle f_1,\dots,f_r\rangle$ の生成系 $\{f_1,\dots,f_r\}$ から I のグレブナー基底 G を計算する方法として，「ブッフベルガーアルゴリズム」が有名であるが，それは 2 重指数的な最悪計算量を持つ．そこで，ブッフベルガーアルゴリズムのある種の逆の操作として，「グレブナー基底」から「非グレブナー基底（の可能性が高い生成系）」を計算するために考案したものが次の定理である．

定理 3 $G=(g_1,\dots,g_t)^\top$ を $k[x_1,\dots,x_n]$ における 0 次元イデアル I のグレブナー基底とする．また行列 $A\in k[x_1,\dots,x_n]^{s\times t}$ に対し，$F=(f_1,\dots,f_s)^\top=AG$ とする．E_t を t 次単位行列とする．次が成り立つ．

1. $\langle F\rangle=\langle G\rangle$ であれば，$s\geq n$.
2. A が $k[x_1,\dots,x_n]^{t\times s}$ において左逆行列を持てば，$\langle F\rangle=\langle G\rangle$.
3. 等号 $\langle F\rangle=\langle G\rangle$ が成立する必要十分条件は，ある $B\in k[x_1,\dots,x_n]^{t\times s}$ が存在して，$BA-E_t$ の各行が G の *syzygy* であることである．

例 4 例 2 のグレブナー基底 $G=(3x+4y-3, y^2+9y)^\top$ に対し，$A=\begin{pmatrix}1 & 0\\ 3y & 1\end{pmatrix}$ とすると，$F=AG=(3x+4y-3, 9xy+13y^2)^\top$ は I の非グレブナー基底．

定理 3 によると，G に左から左可逆行列 A をかけることで，非グレブナー基底（の可能性が高い）$F=AG$ が得られ，G は $\langle F\rangle$ のグレブナー基底になっている．これにより，出力（グレブナー基底）から入力（非グレブナー基底）を得ることができ，大量のデータセットを高速に計算できる．実際，論文 [2] においては，逆順による手法では通常の生成法に比べ 1000 倍以上短い時間でデータセットを生成できている．

5. まとめ

本稿では機械学習を用いたグレブナー基底の計算において，データセットを大量に生成する際の工夫「逆順生成」について紹介した．この逆順生成の手法はグレブナー基底のみならず他の数学の計算についても同様に問題提起することができ，機械学習の視点からの新しい数学の領域が発掘されることも期待される．またデータセットの生成の詳細や，Transformer による学習の精度については，論文 [2] に記載されている．

6. 参考文献

[1] Cox, D. A., Little J. B. and O'Shea, D.: Ideals, Varieties and Algorithms: An Introduction to Computational Algebraic Geometry and Commutative Algebra. Fourth Edition, Undergraduates Texts In Mathematics, Springer, New York, 2015.

[2] Kera, H., Ishihara, Y., Kambe, Y., Vaccon, T. and Yokoyama, K.: Learning to compute Gröbner bases. In the Thirty-eighth Annual Conference on Neural Information Processing Systems, 2024.

[3] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I.: Attention is all you need. In Advances in Neural Information Processing Systems, volume 30. Curran Associates, Inc., 2017.