

動的数学ソフトウェアの教育利用について

福岡大学・理学部 濱田 龍義 (Tatsuyoshi Hamada)
Faculty of Science,
Fukuoka University

1 はじめに

本稿では動的数学ソフトウェアを利用した最近の教育利用事例について述べる。動的数学ソフトウェアを語る上で、原型となった動的幾何学ソフトウェア、もしくは対話式幾何学ソフトウェアと呼ばれるソフトウェアの存在は、欠かすことができない。最初の動的幾何学ソフトウェアは、1980 年代初頭に開発された Geometric Supposer と言われている [1]。また、商業的に成功したソフトウェアとしては、対話式幾何学ソフトウェア Cabri (Jean-Marie Laborde, 1986-), 動的幾何学ソフトウェア Geometer's Sketchpad (Nicholas Jackiw, 1986-)¹ 等が国内外で利用されてきた。

日本国内においては 98/DOS の時代から GC(飯島康之, 1986-) の開発が継続的に行われており、Windows 版, Java 版, HTML5 版と形を変えて、多くの教育実践事例を蓄積されている [2]。また、2003 年に Cinderella (Jürgen Richter-Gebert, Ulrich Koretenkamp, 1998-) が明治大学の阿原一志氏によって紹介されたこともあり、一定の利用者を獲得している [3]。

個人的には、オープンソースソフトウェアとして実装された KSEG (Ilya Balan, 1996-2007) を紹介していたが [4], 残念ながら多くの利用者を獲得するまでには至らなかった。その後、2006 年に GeoGebra の存在に気づき、紹介を行っている [5]。GeoGebra は様々な PC 環境で動作することを目指して Java で実装されていた。ただ、開発当初の計算機の性能では Java アプリケーションを快適に動作させるには不十分であった。しかし、計算機速度の向上とメモリの大容量化により、現在では快適に利用できる。また、北海道教育大学の和地輝仁による日本語化作業も進み、日本国内でユーザを獲得する大きな要因となっている。最近では、Java だけでなく、HTML5 や Javascript による実装も進んでおり、タブレットや携帯端末上での利用も可能となった。

続いて動的数学ソフトウェア GeoGebra を中心に解説し、さらに実際の講義で行っている実践事例について紹介を行う。

2 動的数学ソフトウェア GeoGebra

GeoGebra は、オーストリア、ザルツブルク大学の大学院生であった Markus Hohenwarter が修士課程に在学中の 2002 年頃に開発を始めたソフトウェアプロジェクトであ

¹招待講演者 Paul Libbrecht により、開発が既に停止されたようだという指摘があった。

る。現在、Markus はオーストリア、ヨハネスケプラー大学の教授となり、国際的な開発チームの中心的人物として GeoGebra の開発を進めている。

GeoGebra の開発初期は動的幾何学ソフトウェアと呼んで良いものであったが、関数のグラフ描画、表計算モード、数式処理システム等、様々な機能を実装し、名実ともに動的数学ソフトウェアと呼べる存在となっている。

2.1 初等幾何学

これまでに紹介したもの以外にも動的幾何学ソフトウェアは多数あるが、多くの人にとって、それは初等幾何学のためのツールとして認識されている。ひとくくりに動的幾何学ソフトウェアと言っても、そのインターフェースはソフトウェアによって大きく異なる。例えば、KSEG はマウスの右クリックによって点を入力し、Shift キーなどで点を複数選択することで、線分や直線、円等の幾何学的オブジェクトを描くことができる。例えば、直線を描くためには、まず2点入力する必要がある。また、線分や直線、円などを複数選択し、選択されたオブジェクトによって数学的に定義される新たなオブジェクトを作図することができる。

一方、GeoGebra はメニュー選択形式である。まず、作図したいオブジェクトをメニューまたはボタンから選択し、その後、定義に必要なオブジェクトを入力していく方式を採用している。すなわち、直線を描くためには、まずボタンもしくはメニューから直線を選択し、その後、マウスのクリックによって2点入力する方法である。

初等幾何学のためのツールとして、GeoGebra には様々な機能が用意されている。例えば、角度や線分の長さを表す記号が用意されている。角度を測ったとき、その角度が直角の場合には自動的に直角の記号が挿入される。他にも、図に表されるような記号についてはプロパティのスタイルによって変更が可能である。

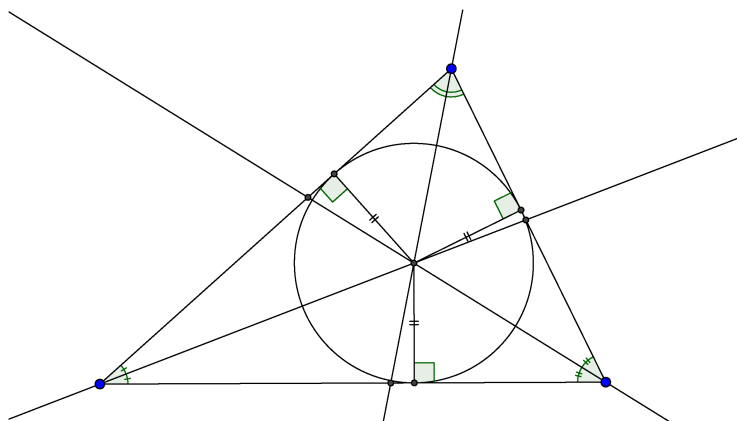


図 1: 三角形の内心

2.2 関数のグラフの描画

動的数学ソフトウェアの中心的な機能が、関数のグラフ描画機能である。Plot[] 等の命令を用いることもなく、 $f(x)=x^2$ のように、計算機における一般的な方法で入力欄に入力するだけでグラフを描画することができる。また、ショートカットキーにより、 $f(x)=x^2$ のような表記にも対応している。Windows の場合には Alt+2, MacOS の場合には control+2 が2乗に対応する。メニューボタンから「点」を選択し、グラフ上をクリックすることで、グラフに拘束された点を作成することもできる。

GeoGebra はフランスで開発された Giac[6] を数式処理エンジンとして採用している。数式処理エンジンによる微分機能を用いることにより、テイラー多項式を動的に可視化することができる。

TaylorPolynomial[<Function>, <x-Value>, <Order number>]

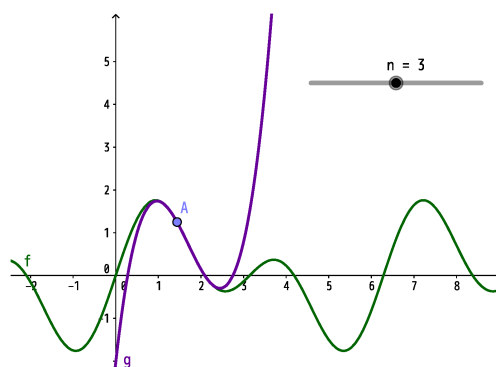


図 2: TaylorPolynomial[sin(x)+sin(2x), x(A), 3]

2.3 ベクトル場と積分曲線

GeoGebra には、ベクトル場を描くための命令も用意されている。以下に示すように、GeoGebra では、一つの命令に対し、引数を複数種類取ることができる。

SlopeField[<f(x, y)>]
 SlopeField[<f(x, y)>, <Number n>]
 SlopeField[<f(x, y)>, <Number n>, <Length Multiplier a>]
 SlopeField[<f(x, y)>, <Number n>, <Length Multiplier a>, <Min x>, <Min y>, <Max x>, <Max y>]

SolveODE[] という命令が用意されており、数式処理システムによって厳密解が得られる場合には厳密解を、そうでない場合には、Runge-Kutta 法を用いて数値積分を行い、与えられた微分方程式に対し、積分曲線を描くことができる。また、始点を指定した際には、動的に積分曲線を生成することができる。

```

SolveODE[<f'(x, y)>]
SolveODE[<f'(x, y)>, <Point on F>]
SolveODE[<f'(x, y)>, <Start x>, <Start y>, <End x>, <Step>]
SolveODE[<y'>, <x'>, <Start x>, <Start y>, <End t>, <Step>]
SolveODE[<b>, <c>, <f>, <Start x>, <Start y>, <Start y'>,
<End x>, <Step>]

```

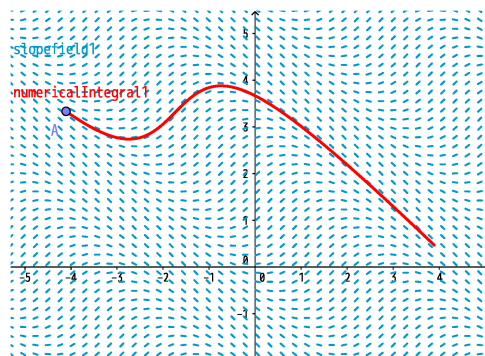


図 3: ベクトル場と積分曲線

2.4 数列の生成

数列の生成は、非常に応用の広い命令である。数値の列を生成することはもちろんのこと、数値によって定義された点等の幾何学的オブジェクトを同時に多数作図することができる。このような構造はリスト構造とも呼ばれる。本稿では紹介していないが、GeoGebra は幾何学的に拘束された点の軌跡を描画することができる。この軌跡の実体は点列によって生成されている。GeoGebra の数列は中括弧によって囲まれたリスト構造を成す。例えば、入力欄に $\{0, 1, 2, 3, 4\}$ を入力すれば、 $0, 1, 2, 3, 4$ という5つの要素を持つリストが得られる。例えば、点を二重ループによって生成すれば、格子点を得られる。また、中括弧は行列の入力にも用いられる。例えば、行列 $\begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}$ を入力するためには、 $\{\{1, 2\}, \{3, 4\}\}$ と入力すれば良い。

```

Sequence[<End value>]
Sequence[<Start value>, <End value>]
Sequence[<Expression>, <Variable>, <Start Value>,
<End Value>]
Sequence[<Expression>, <Variable>, <Start Value>,
<End Value>, <Increment>]

```

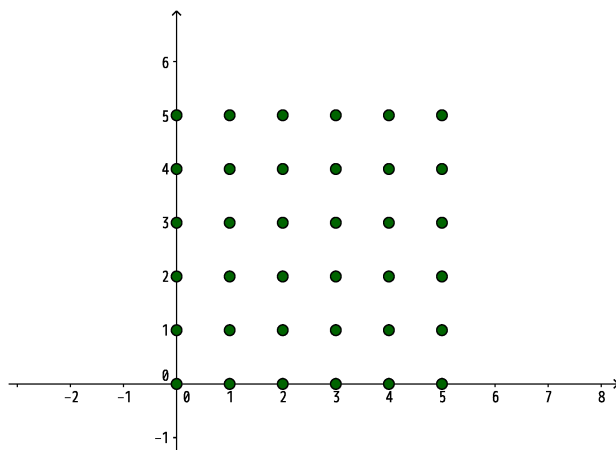


図 4: $\text{Sequence}[\text{Sequence}[(s, t), s, 0, 5], t, 0, 5]$

2.5 表計算

GeoGebra には特定の機能に特化したパースペクティブが用意されている。表計算は、そのひとつで、統計処理やボロノイ図等の計算幾何学にも対応している。

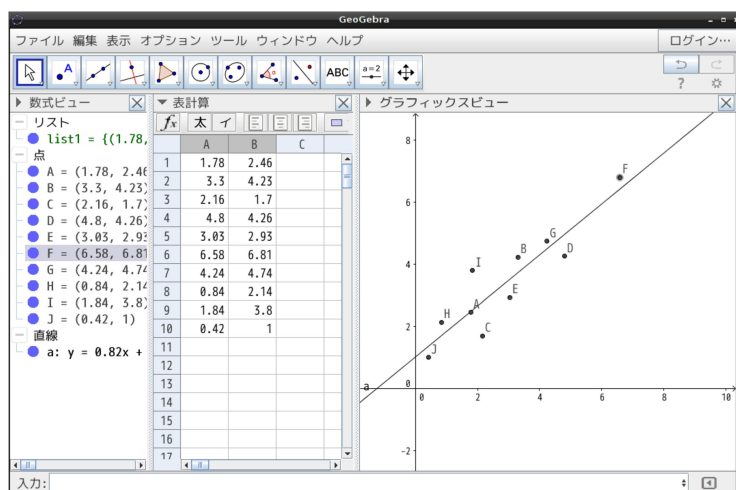


図 5: 点列と回帰直線

いわゆる表計算ソフトと同様に、セルにひとつずつデータを入力することもできるし、等差数列に対する補完等にも対応している。ただし、若干注意が必要である。例えば、等差数列の第 i 項から第 $i+n$ 項まで与えられていたとする。一般に、表計算ソフトは第 i 項から第 $i+n$ 項が入力されているセルを選択したあと、選択したセルを拡張すると、第 $i+n+1$ 項以降が自動入力される。しかし、いまのところ、GeoGebra の表計算は、このような機能に対応していない。第 i 項と第 $i+1$ 項の 2 つのセルを選択した時だけ拡張可能である。また、CSV 形式のデータ入力にも対応している。

2.6 数式処理

GeoGebra は Giac という数式処理システムを内包している。Giac はフランス、ジョセフフーリエ大学の Bernard Parisse によって 2000 年から開発が進められている数式処理システムである。C++ で記述されており、ソースコードは 15 万行を越える。Giac は Emscripten によって JavaScript に変換され、GeoGebra に内包される [7]。GeoGebra の一般の入力とは異なり、CAS で関数を定義する際には $:=$ を用いる。

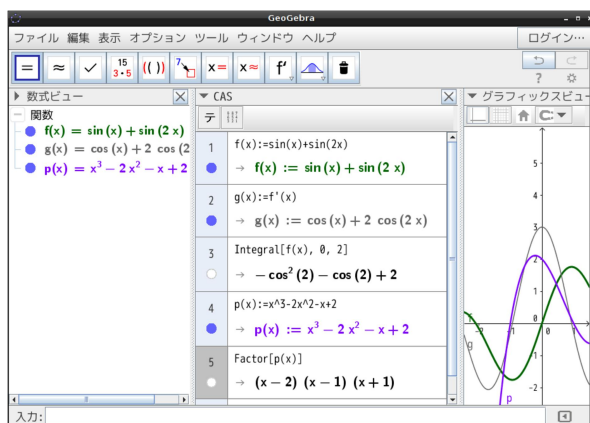


図 6: CAS

2.7 複素平面

GeoGebra は複素平面上での描画にも優れている。平面上の点を複素平面上の点として扱うことができ、また、複素平面上の点を入力するためのメニューボタンも用意されている。単位円上の点を複素函数によって写し、その軌跡を描いて確認したり、複素級数（実際には有限和しか取れない）の収束の様子を観察することも可能である。

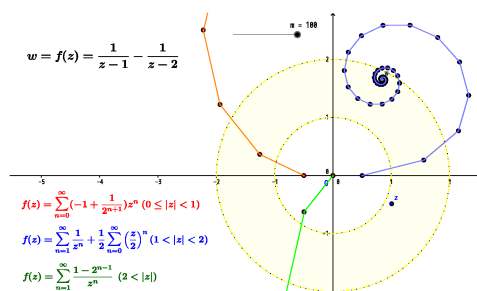


図 7: ローラン級数の収束性

2.8 3D グラフィックス

3D グラフィックス機能は、長いベータ期間を経て 2014 年 9 月に正式に実装された機能である。フランスの数学教員である Mathieu Blossier が中心となって実装を行った。正多面体の描画や、展開図の作成に対応しており、展開図を印刷してペーパークラフトを作成することもできる。

また、2 変数関数のグラフや、媒介変数表示による曲面の描画にも対応しており、数式処理機能と組み合わせることで、ベクトル解析学や初等的な微分幾何学の講義に用いることもできる。残念ながら現在のところ、多項式の陰関数表示による曲面の描画には対応していないが、Google Summer of Code²において、インドの Shamshad Alam による実装が採用されており、将来、新機能として追加される可能性が高い [8]。

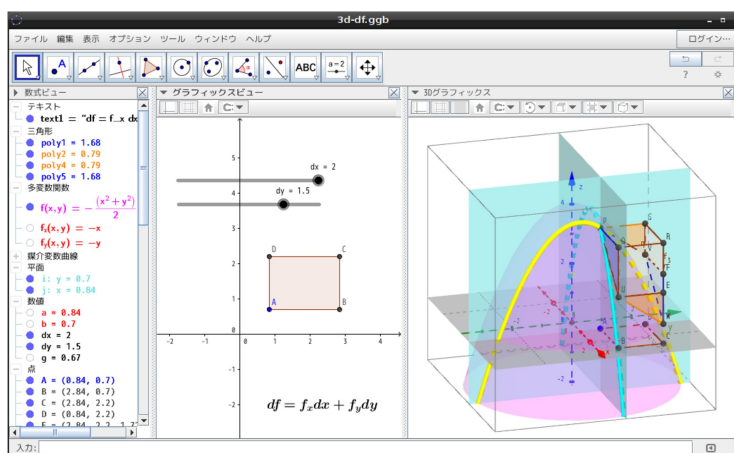


図 8: 放物面と全微分

2.9 画像ファイルの出力

GeoGebra は様々な画像ファイルに対応しており、メニューの「ファイル」→「エクスポート」→「グラフィックスビューを画像として (png, eps) ...」を選択することで、PNG, PDF, EPS, SVG, EMF 等の形式に出力することができる。また、スライダーと組み合わせてアニメーション GIF への出力も可能である。これまで、3D グラフィックスビューについては、直接、画像ファイルを出力することができず、クリップボードを介する必要があったが、最近になって画像ファイルへの出力についても対応した。

最近では、 $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ に画像ファイルを埋め込む際は、EPS でなく、PDF が推奨されるようになったが、GeoGebra で PDF を生成する際は注意が必要である。現在、GeoGebra が生成した PDF は extractbb を実行する際にエラーとなる。これを防ぐためには、GeoGebra で生成した PDF ファイルをテキストエディタなどで開き、trailer と書かれた行のすぐ上にある空行を取り除けば良い。

²Google によって主催されているイベントで指定されたオープンソースのプロジェクトで、夏の間に課題をクリアした学生に賞金を支払う制度。

2.10 $\text{T}_\text{E}\text{X}$ 形式の出力

GeoGebra は PDF 形式の出力に対応しているだけでなく、様々な $\text{T}_\text{E}\text{X}$ マクロによる作図パッケージや作図言語に対応している。「ファイル」→「エクスポート」→「グラフィックスビューを PSTricks へ」というメニューが用意されている。他にも PGF/TikZ, Asymptote にも対応している。今のところは、簡単な平面図（空間図形は出力できない）に限られており、PGF/TikZ でエクスポートして、 $\text{PDF}\text{\LaTeX}$ で組版を行う方法が一番無難である。なお、出力される PGF/TikZ のソースコードは単体でコンパイルできる形式になっており、実際に原稿等に埋め込む際には編集が必要である。

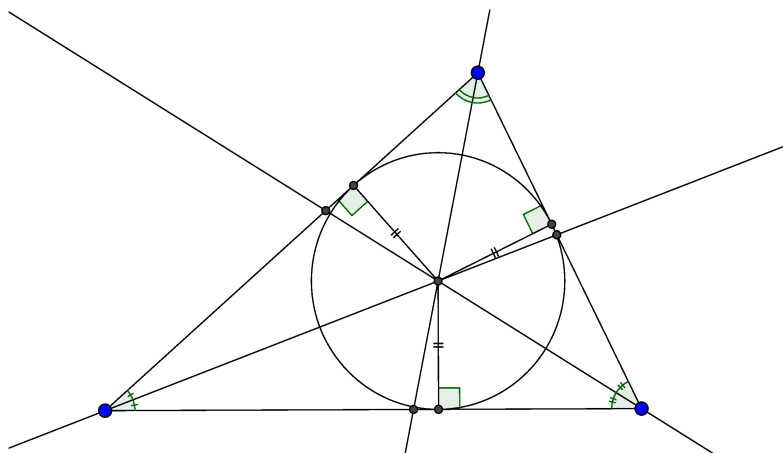


図 9: 図 1 を PGF/TikZ で出力

2.11 教育素材の共有

教材の作成時間の確保は悩ましい問題であるが、世界中で GeoGebra 教材を共有しようという動きがある。GeoGebra Tube と呼ばれるサービスでは 30 万個以上の教材が公開されている。その殆どが欧米の言語圏の教材であるが、少しずつではあるが日本語の教材も増えつつある。

GeoGebra Tube への教材提供にあたっては GeoGebra 本体とサービスへのログインアカウントが必要である。「ファイル」→「エクスポート」→「動的なワークシートを Web ページとして (html) ...」を選択する。あとは指示に従って、タイトルや説明を追加すれば良い。今後、日本国内で多くのユーザを獲得するためにも、GeoGebra 本体と同様に、Web サービスや教材の日本語化についても、早急に取り組まないといけない。

3 動的教材と静的教材

ここまで、動的数学ソフトウェア GeoGebra の基本機能について紹介してきた。前章の終わりでも述べたが、GeoGebra は $\text{T}_\text{E}\text{X}$ 形式で出力が可能である。 $\text{T}_\text{E}\text{X}$ 形式で出力さ

れたファイルをタイプセットして印刷することで、印刷物による教材を作成する際にも便利である。この章では GeoGebra などの動的ソフトウェアによる教材、すなわちマウスやタブレットを用いて任意の点や図形を動かすような教材を動的教材と呼び、印刷物による教材を静的教材と呼ぶことにする。

動的教材の良さは、対話的に図を変形できる点である。また、表計算を利用して、統計の問題を探索することも可能である。一方で、教育の現場で1つの授業、講義を全てコンピュータの操作に充てるべきかと言えば、否定する方も多い。タブレットや携帯電話の教育利用も検討する価値があるが、それぞれ、一長一短がある。東邦大学の高遠節夫を中心とするグループによって開発されている KETpic[9] の図入り印刷教材等も静的教材の一種である。KETpic や PGF/TikZ 等を用いることで、正確な図の作成が可能となる。定規を用いて図を計測したり、函数のグラフ上の点をデータを用いて描画する場合に威力を発揮する。残念なことだが、かなりの数の学生が、データから正確なグラフを描画する訓練が足りていないのが現状である。方眼紙を用いれば良いという考え方もあるが、教員が、誘導のための図を自由に作成して、示すことができることを考えると、GeoGebra や KETpic 等を作図ツールとして活用することは十分に有効である。

最近では $\text{T}_{\text{E}}\text{X}$ と既存の言語を組み合わせる動きも活発であり、教材作成のツールとして今後に期待している。言語の候補としては Lua[10] や Python[11] 等が有名である。Lua はブラジルのリオデジャネイロ・カトリック大学のコンピュータ科学科で開発された手続き型言語である。高速な動作と、高い移植性、組込みの容易さが特徴である。Lua $\text{T}_{\text{E}}\text{X}$ は、pdf $\text{T}_{\text{E}}\text{X}$ の後継として開発されており、 $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ ソースコードの中に Lua ソースコードを埋め込むことができる。現在、次世代の $\text{T}_{\text{E}}\text{X}$ 日本語実装系として、北川弘典を始めとするプロジェクトチームによって Lua $\text{T}_{\text{E}}\text{X}$ -ja[12] の開発が進められており、TeX Live[13] 2014 以降にも収録されている。ただ、Lua $\text{T}_{\text{E}}\text{X}$ 自体の仕様が確定しておらず、組版が遅いという欠点がある。

その他には、Python $\text{T}_{\text{E}}\text{X}$ [14] が知られている。Python $\text{T}_{\text{E}}\text{X}$ は Python だけでなく、Ruby[15] や Octave[16], Julia[17] 等のコードを埋め込むことができる。Python $\text{T}_{\text{E}}\text{X}$ は、Lua $\text{T}_{\text{E}}\text{X}$ とは異なり、エンジン拡張ではないので、 $\text{T}_{\text{E}}\text{X}$ エンジンとして p $\text{T}_{\text{E}}\text{X}$ 等を選択することも可能である。また、オープンソース・ソフトウェアの数式処理システム Sage のコードを埋め込める Sage $\text{T}_{\text{E}}\text{X}$ [18] が Dan Drake によって開発が進められており、TeX Live 2009 から収録されている。

個人的には、マークシート方式の選択式試験問題の作成と管理を容易にするパッケージ Auto Multiple Choice[19] にも注目しており、現在、検討中である。

4 動的数学ソフトウェアの可能性

動的数学ソフトウェアも、動的幾何学ソフトウェアも、ともすれば、初等幾何学だけのためのツールと捉えがちである。本稿で紹介した実践例を見れば明らかなように、動的数学ソフトウェアは数学的な発想を表現するための環境として捉えることができる。

数学上の概念を基盤として、適切に実装されたソフトウェアはしばしば、作者の意図以上の可能性を秘めている。利用者は、どうしても決められた枠内で考えがちであるが、

ソフトウェアはもっと自由なものであり、様々な数学分野を横断的に繋げることができると考えている。ただ、数学は広大で深淵なものであるなので、現在のところソフトウェアで表現できるのは、ごく一部に過ぎない。それでも、「まず疑ってみる」ことが数学ソフトウェアの利用者として求められる姿勢ではないだろうか。また、数学ソフトウェアを用いることで、利用者に数学上の概念を誤解させてはいないか、さらなる検証が必要である。

我々は、数学ソフトウェアを数学の一分野と考えている。また、現在、数学ソフトウェアは数学者や、数学教育者、数学利用者、そして、数学を苦手と感じる人たちとを結びつけるコミュニケーションツールとしての役割を果たしている。

一方で、数学をソフトウェアとして開発、実装することは、決して簡単とは言えない。また、現在の計算機システム上で数学ソフトウェアを利用するためには、計算機の基本的な仕組み、メモリの利用や変数の型など、最低限の知識が必要不可欠である。大学教育における数学ソフトウェアの利用は今後も増えていくと思われるが、数学を学ぶ学生のために必要な計算機システムの知識は、どのようなものであるべきか、まだまだ合意が形成されているとは言えない。さらなる議論を重ねる必要があると考えている。

参考文献

- [1] Shwartz, Yerushalmy and Wilson, “The Geometric Supposer : What is it a Case of?”, Lawrence Erlbaum Associates, 1993.
- [2] 飯島康之, 愛知教育大学飯島研究室, <http://www.auemath.aichi-edu.ac.jp/teacher/iijima/iijima.htm>
- [3] J. リヒターゲバート, U.H. コルテンカンブ著, 阿原 一志訳, 「シンデレラ 幾何学のためのグラフィックス」 Springer-Verlag Tokyo, 2003.
- [4] 濱田龍義, “Pedagogical practice cases of some trajectories with dynamic geometry software”, 数理解析研究所講究録 第 1865 巻 2013 年 185–189.
- [5] 濱田龍義, 「大学初年級における GeoGebra の教育利用 (数式処理と教育)」, 数理解析研究所講究録 第 1674 巻 2010 年 112–119.
- [6] B. Parisse, <https://www-fourier.ujf-grenoble.fr/~parisse/giac.html>
- [7] Z. Kovács and B. Parisse, Giac and GeoGebra: improved Gröbner basis computations, <https://www.ricam.oeaw.ac.at/specsem/specsem2013/workshop3/slides/parisse-kovacs.pdf>
- [8] Google Summer of Code, <https://dev.geogebra.org/trac/wiki/Gsoc2015>
- [9] 山下哲, 高遠節夫, KETpic による教材作成と Symbolic Thinking, 数理解析研究所 第 1780 巻 2012 年 72–82.

- [10] The Programming Language Lua, <http://www.lua.org/>
- [11] Python.org <https://www.python.org/>
- [12] Lua_{TEX}-ja プロジェクト wiki, <https://osdn.jp/projects/luatex-ja/wiki/FrontPage>
- [13] TeXLive - TeX Users Group, <https://www.tug.org/texlive/>
- [14] Geoffrey M Poore, “PythonTeX: reproducible documents with LaTeX, Python, and more”, Computational Science & Discovery, Volume 8, Number 1.
- [15] オブジェクト指向スクリプト言語 Ruby, <https://www.ruby-lang.org/ja/>
- [16] GNU Octave, <https://www.gnu.org/software/octave/>
- [17] The Julia Language, <http://julialang.org/>
- [18] Sage_{TEX} Embed Sage code and plots into L_AT_EX, <https://www.ctan.org/tex-archive/macros/latex/contrib/sagetex/>
- [19] AMC - 選択式試験問題の管理と自動採点, <http://home.gna.org/auto-qcm/>