実践報告

Google Colaboratory を活用した高等学校物理の授業実践

― ケプラーの第三法則 ―

＊能代谷　賢治，＊内山　哲治

Practical Report Using Google Colaboratory in High School Physics Class
- Kepler's Third Law -

NOSHIROYA Kenji and UCHIYAMA Tetsuji

要　旨

高等学校物理にプログラミングを導入すべく，プログラミング言語の1種である Python を無料で OS や端末を問わずに実行できるサービス“Google Colaboratory（Colab）”を検討した。生徒自らがプログラム構造を考え，ケプラーの第三法則を導出するプログラミング教材を開発し，授業実践を行った。本実践において，高等学校物理へのプログラミング導入には2つの困難があることが分かった。1つ目は生徒の学習活動における情報端末使用の難しさであり，2つ目はテキスト言語におけるプログラム構造の視認性の低さである。これらの困難は，操作マニュアルの事前作成・共有およびプログラムを機能別に分割することによって解決できた。

Key words：Google Colaboratory，Python，ケプラーの第三法則，プログラミング教育，GIGA スクール構想

1．はじめに

1-1．研究背景

平成29・30年の学習指導要領改訂に伴い，小・中・高等学校におけるプログラミング教育の充実が求められている。このプログラミング教育は，全ての学習の基盤となる資質・能力の1つである「情報活用能力」の育成に必要不可欠である。この「情報活用能力」とは，世の中の様々な事象を情報とその結び付きとして捉え，情報及び情報技術を適切かつ効果的に活用して，問題を発見・解決したり自分の考えを形成したりしていくために必要な資質・能力とされている（文部科学省，2020）。高等学校のプログラミング教育は主として情報科が担っている。平成30年告示の高等学校学習指導要領においては「情報Ⅰ」が必修科目として新設された。「情報Ⅰ」では，プログラミングやネットワーク，情報セキュリティ，データベースの基礎的な内容が扱われる（文部科学省，2019a）。文部科学省は，この「情報Ⅰ」の担当教員の指導力向上を目指し，教員研修用教材を公開している（文部科学省，2019b）。この教材の中では，プログラミング言語である Python を用いたプログラムが例示されている。

プログラミング教育の充実は，各学校における情報端末の整備によって支えられている。公立の小・中学校では令和2年度以降，GIGA スクール構想の実現に向け児童生徒に対し1人1台の情報端末が貸与されている。しかしながら，小・中学校で貸与された情報端末は各自治体で異なっており，その情報端末の OS は

＊　宮城教育大学教職大学院

3種類（Chrome OS，Windows OS，iPad OS）に大別される（文部科学省，2021）。つまり，情報端末のOSは学校間において乱立状態にあると言える。貸与されている情報端末は各学校内で一括管理されている場合が多く，アプリケーション等のインストールも容易ではない。高等学校では令和６年度に１人１台の情報端末環境の整備が完了される予定である（文部科学省，2022）。その環境整備は各自治体の公費で進める場合や，各自治体または学校が情報端末を指定し，学校に持ち込ませる場合（BYAD:Bring Your Assigned Device），生徒が自由に用意した端末を学校に持ち込ませる場合（BYOD:Bring Your Own Device）がある。各自治体の公費で進める場合やBYADの場合，情報端末のOSは学校内で統一される。しかし，BYODの場合は，学校内においてもOSの乱立状態が生じうると予想される。

プログラミングは，反復処理や条件分岐処理などを基本の構文とし，それらを論理立てて組み合わせることによってコンピュータに自らが意図した処理を行わせるものである。一方，物理などの自然科学は自然界に見られる現象の原因や仕組みを，仮説を基に論理立てて解明する学問である。物理とプログラミングは，基本となる構文や仮説を“論理的に積み上げていく”という構造において親和性が高いと言える。高等学校物理においては，これまでプログラミング言語を用いた物理シミュレーション教材の開発や教育実践が行われてきた（加藤，2004;池口・内山，2009;山口・内山，2013;竹ヶ原･内山,2015;奈良,2020）。物理シミュレーション教材は，重ね合わせの原理による合成波の描画や原子の熱運動など，目に見えない物理現象に対する生徒の視覚的な理解を促す教材として有効であると考えられる。しかしながら，これまで開発・実践されてきた物理シミュレーション教材はプログラミングを得意とする教員によって作成され，授業実践時にはそのプログラム構造がブラックボックス化されていることが多い。その際，生徒はそのプログラム構造を殆ど目にすることはできず，与えられたシミュレーション教材の表層ばかりに注目することが予想される。つまり，生徒の学びが受動的になる可能性が高いと思われる。

1-2.　研究目的

上述した情報端末の整備状況および物理とプログラミングの親和性の高さを背景に，生徒の主体的な学びを実現させるためには，どのOSでも活用することができ，生徒自身が論理展開を考えることのできるプログラミング教材の開発・実践が求められる。本研究は，高等学校物理へプログラミングを導入し，物理の学習内容を生徒に深く考えさせる授業を検討することを目的とした。

２．研究方法

2-1.　プログラミング言語の選定

プログラミング言語はソースコードの表現方法の違いによって，ビジュアル言語とテキスト言語の2つに大別される。本節では，それぞれの言語の特徴と代表的なプログラミング言語について述べる。

ビジュアル言語は，ブロックやアイコンなどの視覚的なオブジェクトを，線などで繋ぐことでプログラムを作成する言語である。ビジュアル言語の中にもいくつか種類があるが，ここではブロック同士を直接繋ぐ言語を“ブロック型”，アイコン同士を線で繋ぐ言語を“フロー型”と表現する。ブロック型はプログラム構造が直線的で理解しやすい特徴がある。具体例として，ScratchやLINE entry等が挙げられる。一方，フロー型は条件を分岐させる等，より複雑なプログラム作成が可能であるという特徴がある。具体例として，LabVIEWやembot等が挙げられる。ブロック型に分類されるScratchのプログラムと，フロー型に分類されるLabVIEWのプログラムの例をそれぞれ図1に示す。これらのビジュアル言語はプログラム構造を視覚的に認識しやすい。この利点を背景として，ビジュアル言語は小・中学校のプログラミング教育で多く用いられている。

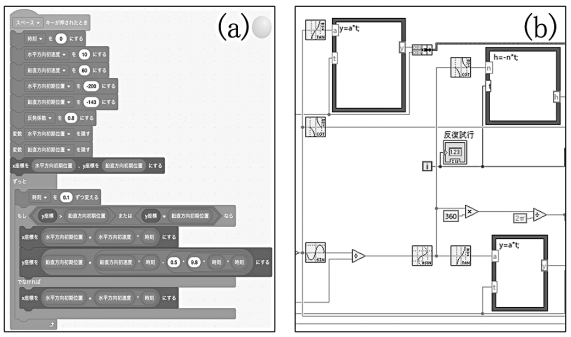

図１　ビジュアル言語のプログラム構造
(a) ブロック型:Scratch，(b) フロー型:LabVIEW

一方，テキスト言語はプログラムを文字や数字，記号のみを用いて記述する言語である。具体例として，Java や Processing，Python 等が挙げられる。テキスト言語に分類される Processing と，Python のプログラムの例をそれぞれ図2に示す。これらのテキスト言語は複雑な計算処理を行うことができ，アプリケーション開発や機械学習など，世界中の様々な分野で実用化に至っている。そのため，ライブラリ（汎用性の高い複数のプログラムのまとまり）が充実しているという特徴がある。しかしながら，岡本ら（2020）は高校生の Python 学習に関するアンケート調査において，90%以上の生徒が Python の学習が難しかったと回答したことから，高校生の Python 学習に関して何らかの困難があることを指摘している。

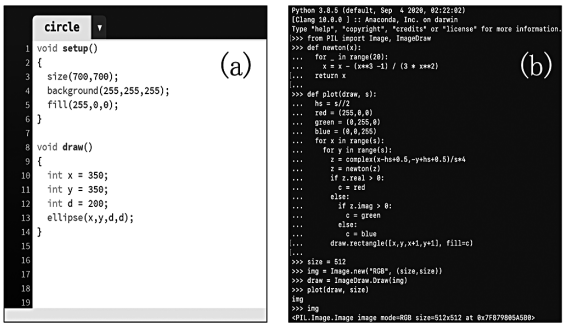

図2　テキスト言語のプログラム
(a)Processing，(b)Python

今回，我々は高等学校物理へのプログラミング導入を検討するに当たり，様々なプログラミング言語の中から本実践で使用するプログラミング言語の選定を行った。ここでは，これまでの物理シミュレーション教材開発で使用されている言語（Scratch，LabVIEW，Java，Processing）および上述した教員研修用教材に例示されている Python に着目した。高等学校物理でのプログラミングの導入を見越すと，使用するプログラミング言語は無料で扱えることや環境構築が簡易であること，複雑な計算処理に対応していることが望まれる。よって，本研究ではこれらのプログラミング言語を，①価格，②環境構築，③複雑な計算処理対応，の3観点によって総合的に判断し，使用するプログラミング言語の選定を行った（表1）。

表1に示すように，ビジュアル言語の Scratch は無料で扱うことができ，小・中学校のプログラミング教育で多く用いられている。また，Scratch は Web ブラウザ上で使用できるオンライン版があり，情報端末にインストールするなどの環境構築が不要となる。しかし，Scratch はブロック型であることから複雑な計算処理が困難である。LabVIEW は理工系計測・制御用のビジュアル型プログラミング言語であり，複雑な計算処理が容易である。しかし，LabVIEW は有料であり，情報端末へインストールする必要があることから教育現場への導入は難しいと考えられる。テキスト言語の Java および Processing は無料で扱うことができ，複雑な計算処理が容易である。しかしながら，情報端末へのインストールが必要であり，科学計算やグラフ描画に必要なライブラリをもインストールする必要がある。Python は無料で扱うことができ，複雑な計算処理が容易であるテキスト言語である。情報端末へのインストールおよびライブラリのインストールが必要であるが，Web ブラウザでも使用できるサービスを用いることによってこれらの環境構築が不要となる。以上より，高等学校物理へのプログラミング導入には Python が適していると考えた。

表1　プログラミング言語の特徴
（一部不要）:一部のサービスでは環境構築が不要となる
△:困難である　　○:可能である

	ビジュアル言語		テキスト言語		
	ブロック型	フロー型			
	Scratch	LabVIEW	Java	Processing	Python
価格	無料	有料	無料	無料	無料
環境構築	必要 （一部不要）	必要	必要	必要	必要 （一部不要）
複雑な 計算処理	△	○	○	○	○

2-2.　Google Colaboratory（Colab）

Google Colaboratory（Colab）は，Google 社が機械学習の教育・研究を目的として開発した，Web ブラウザ上でテキスト言語の Python を実行できるサービスである。なお，Google アカウントへのログインが必要となる。Colab を教育現場で使用するメリットは大きく3つ存在する。1つ目は，「無料で，端末を問わず活用できること」である。Colab は主要な Web ブラウザ（Chrome，Edge，Safari 等）上で Python を実行するサービスであり，情報端末にインストールする必要がなく，また，情報端末の CPU 処理能力にも依存しない。したがって，プログラミングの環境構築も不要になる。2つ目は，「プログラムの保存・

共有が容易であること」である。Colab はログインした Google アカウントと連係し，作成したプログラムは Google ドライブに自動で保存される。また，プログラムを他のユーザと簡単に共有でき，共同編集することもできる。3つ目は，「ライブラリが充実していること」である。Colab は Python を実行するため，Python のプログラムをほぼそのまま活用できる。つまり，Python の豊富なライブラリを活用することができる。また，数値計算を行う NumPy や，グラフを描画する matplotlib などのライブラリは，既に Colab にインストールされている。

2-3.　Colab を活用するプログラミング物理教材の開発

本研究では Colab を活用する Python プログラムを作成するにあたり，ケプラーの第三法則の導出を取り上げた。ケプラーの法則は，ケプラーが彼の師であるティコ・ブラーエの惑星観測データを解析し発見した3つの法則であり，後のニュートンが発見した万有引力へつながる重要な法則である。しかしながら，ケプラーの第三法則は観測値による裏付けもなしに示されている（山本，2014;Dreyer，1953）。また，渡辺（1991）によれば，ケプラーの第三法則は膨大な数値計算によって発見されたと予想されるが，ケプラーはその第三法則を発見する過程についての手がかりを何も残していない。われわれはこれらの歴史的背景を踏まえ，生徒にケプラーの第三法則の発見過程を追体験させることを意図した教材を開発した。

本教材は太陽系惑星の観測データ（軌道長半径aと公転周期T）から，ケプラーの第三法則（$T^2=ka^3$ k:定数）をプログラミングによって見出す教材である。本教材の特徴は，生徒がプログラムを殆ど記述することなくケプラーの第三法則を導出することができる点にある。これにより，生徒のコーディング能力（プログラムを記述する能力）ではなく，論理的思考力（どのプログラムをどの順序で実行すべきかを考える力）の育成に重点を置いた授業展開が可能となる。本教材およびサンプルプログラム，本教材の Colab 操作マニュアル，デモンストレーション動画は図3に示す QR コードおよび参考文献からダウンロードできる。本教材のプログラムコードおよび操作方法の詳細は付録に示す。

図３　本教材の QR コード

2-4.　授業実践

我々は，令和3年度に宮城県内の高等学校において授業実践を行った。この高等学校では令和3年度から1人1台の情報端末環境が整備されており，各教科の授業や探究活動，学校行事などの様々な教育活動において Chromebook（ChromeOS が搭載された情報端末）が活用されている。対象は普通科理系第2学年の生徒であり，物理を科目選択している生徒であった。本実践は高等学校物理の万有引力単元で行い，全3時間のうち2時間目であった。

以下では，本実践の授業展開および実践内の働きかけの工夫を述べる。

導入では，最初に単位の異なる2種類の観測データ（a〔AU:天文単位〕とT〔year〕，a〔m〕とT〔day〕）および6つのテキストファイル（①近似直線の算出，②ライブラリの読み込み，③データの読み込み，④グラフの作成・表示，⑤データの処理，⑥近似直線のR^2値（決定係数）の算出）をGoogle Classroom（Google 社が提供する授業支援ツール）を用いて生徒と共有し，「Classroomで共有されたファイルを全てダウンロードしなさい。」と指示した。その後，予め作成しておいたColab操作マニュアルを生徒と共有した。

展開は4つの段階に大別される。第1段階は，生徒がPythonなどのテキスト言語で求められる論理展開を理解できているかどうかの判断過程とした。第1段階では a〔AU〕とT〔year〕の観測データを使用し，データプロットと近似直線を示すグラフを作成するプログラム実行順序を生徒に考えさせた。その際，「6つのプログラムの実行順序を考え，データの点と近似直線を示すグラフを表示させなさい。」と生徒に対して指示し，図4に示すT-aグラフを生徒に表示させた。われわれは，Microsoft ExcelやGoogle スプレッドシートなどの表計算ソフトとPythonとでは，データプロットと近似直線を示すグラフの作成処理順序が異なっており，教育現場で表計算ソフトの使

用に馴染みのある場合は，Python等のテキスト言語を扱う場合に注意が必要になることを指摘している（能代谷・内山，2021）。図4に示すようなデータプロットと近似直線を示すグラフの作成処理において，Microsoft Excel等の表計算ソフトではデータプロットを散布図で表示し，可視化された散布図に近似直線およびその関係式を追加するという手順を踏む。一方，Python等のテキスト言語では散布図を作成するためのコード，近似直線の傾きや切片を算出するコードおよび算出した近似直線を描画するコードを実行した後，散布図および近似直線をまとめて表示するプログラムを実行する。つまり，Python等のテキスト言語では散布図が可視化されていない状態で近似直線を追加する必要がある。本実践においても，対象生徒は表計算ソフトの使用に馴染みのあることが予想された。そのため，まず生徒にプログラム実行順序を考えさせ，グラフを表示させた。その後，テキスト型言語で求められる“全ての計算処理を行ってから表示する”という論理展開を補足した。

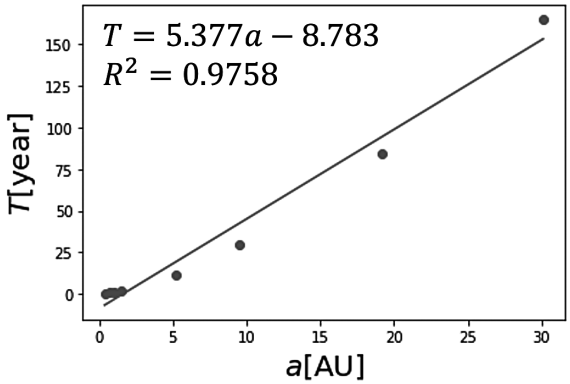

図４　データの演算を行なっていない *T-a* グラフ（関係式は後から追加）

第２段階は，生徒にデータの演算を行わせ，ケプラーの第三法則を導出することを目的とした。まず，生徒にデータの演算を行わせ，図５に示す T^2-a^3 グラフを生徒に表示させた。この時，生徒に対しては，「R^2 値は決定係数と言います。この値はデータと近似直線の当てはまり度合いを０以上１以下で表しており，１に近いほどデータと近似直線がよく当てはまっていることを示しています。そこで，この R^2 値を可能な限り１に近づけるよう，データに演算をしなさい。」と生徒に指示した。この指示は，生徒にどのような演算をすべきかを敢えて伝えず，R^2 値を指標として生徒自身に多様な演算内容を考えさせることを意図して行った。第２段階として設定していた時間の後半では，演算内容を１から５までの自然数の累乗に限定し，生徒の T^2-a^3 グラフの作成を誘導した。

生徒に T^2-a^3 グラフを表示させた後，算出された近似直線の関係式からケプラーの第三法則を導出した後，生徒に「ケプラーの第三法則の定数kがほぼ１になる。」という教科書の記述を確認させた。このタイミングで生徒に教科書を確認させることにより，生徒の納得感のある理解を促した。

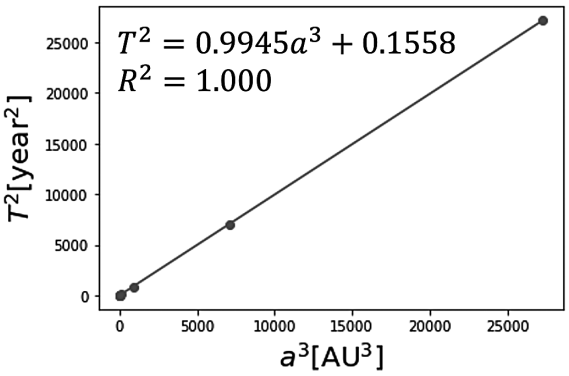

図５　T^2-a^3 グラフ（関係式は後から追加）

第3段階は，異なるデータを用いて分析させ，得られる結果について考察することを目的とした。単位の異なる観測データ（a〔m〕とT〔day〕）を用い，生徒に第2段階と同一のプログラム順序で分析させた。このデータの分析により，図6に示すグラフおよび近似直線の関係式が得られる。生徒が分析した後，その分析結果から第2段階同様の手順でケプラーの第三法則を

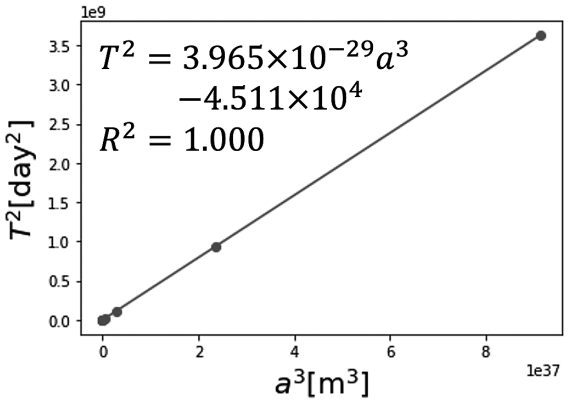

図６　単位の異なる観測データを用いた T^2-a^3 グラフ（関係式は後から追加）

導出し，生徒に2つの導出結果を比較させた。この単位の異なる観測データを用いた分析・比較は，“定数kがほぼ1になる”という教科書の記述を納得して受け入れた生徒の思考を揺さぶることを目指した働きかけであった。

第４段階は，定数kがほぼ１になるという教科書の記述を考察させることを目的とした。生徒にケプラーの第三法則の定数ｋがほぼ１になる原因を考えさせ，生徒に“地球を基準としている”ことを気づかせた。この“地球を基準としている”ことは，aの単位に“天文単位（地球と太陽の距離を１とする単位）”が用いられ，Tの単位に“年（地球の公転周期をほぼ１とする単位）”が用いられていることに起因する。

３．本実践における生徒の実態と考察

本実践における生徒のプログラム実行結果および生徒の様子から，本教材及び本実践の授業展開の有効性を検討することとした。本実践の記録方法は以下の通りである。実践時は授業の様子を教室後方から録画し，黒板と表示した授業スライド，授業中の音声を記録した。また，生徒全体に対して説明や指示を出した後，机間指導を行い生徒のプログラム実行結果や生徒同士の対話を観察した。机間指導中は，その観察内容を手書きによって記録した。

本実践における生徒の様子を述べる。導入では，「Classroomで共有されたファイルを全てダウンロードしなさい。」と生徒に指示したところ，何名かの生徒からはダウンロードしたファイルの保存先を見失う，ショートカットキーを扱うことができず，２種類の観測データと６つのテキストファイルを１つずつダウンロードする等の様子が確認された。実践前にはファイルダウンロードの時間を３分程度と見込んでいたが，実際には10分程度かかった。これは，我々が生徒の情報端末の使用に関する実態を把握できておらず，上述した指示が生徒にとって不明瞭であったためと考察した。

展開の第１段階では，生徒は事前に共有されたColab操作マニュアルを参照しながらショートカットキーを確認したり，次の操作を確認したりといった，Colab上の操作を進めることができていた。これより，事前に作成されたColab操作マニュアルは生徒が授業中に分からなくなってしまった時の手助けとして有効であったと判断できる。加えて，生徒がColab操作マニュアルに示されたショートカットキーを用いることによって，生徒の考える時間が増えるといった授業時間の有効活用にも寄与していたのではないかと推察される。また，生徒は６つに分割されたプログラムの実行順序を考え，試行錯誤しながらグラフを作成することができていた。一部の生徒はPythonで求められる論理展開として正しくないプログラム実行順序で実行し，Colab上でエラーが表示されていた。しかしながら，その生徒らは実行順序をもう一度考え，実行してみるなどの試行錯誤を繰り返しながら，最終的にグラフを描画することができていた。これより，Python等のテキスト言語は，プログラムを分割し実行順序を考えさせる方法を用いることで，生徒自身が考えるプログラミング活動に取り入れられることが分かった。

第２段階において，「R^2値を可能な限り１に近づけるように，データに演算をしなさい。」と生徒に指示したところ，生徒はまず加法や減法による演算を試みていたが，グラフの概形及びR^2値が変化しないことに気づいた。その後，乗法や除法，累乗による演算に自然と移行していった。その際，班内で演算内容を話し合ったり，役割分担（例:Tの演算を２乗で固定し，aの演算を２乗から５乗までを４人で分担して一斉に実行してみる）を行ったりしながら，R^2値を可能な限り１に近づけようと試行錯誤していた。この指示には，生徒の視点を演算内容からR^2値へとずらす作用があり，R^2値が１に近い方が良いという答えを提示しながらも，演算内容は生徒自身に考えさせる効果があったと考えられる。その後，分析結果から定数kがほぼ１になるケプラーの第三法則を導出し，生徒に教科書の記述を確認させたところ，生徒は教科書の記述を納得して受け入れている様子であった。

第３段階において，生徒はColabの扱いに徐々に慣れてきた様子であり，Colab上の操作が円滑になりつつあった。第２段階と同様の手順で導出を行ったところ，生徒は定数kが3.965×10^{-29}となることに着目していた。その際，生徒は定数kがとても小さい数であり０に近似できるのではないかと考えていたり，定数kはほぼ１でなくてもよいのではないかと考えていたりしていた。ここで，定数kがとても小さい数であ

り０に近似できるのではないかと考えていた生徒に対しては，数値だけを見ると小さな値だが，観測データに基づく有限な値であることを補足した。第３段階では，単位の異なる観測データを同一プログラムによって分析し，その結果を比較させた。これにより，生徒は第２段階において納得して受け入れた教科書の記述を吟味することができたと考えられる。

第４段階において，生徒は提示された２種類の観測データを比較し，単位の違いに着目していた。そのため，天文単位の定義や天文単位を用いている意味を問うたところ，生徒は天文単位が地球と太陽の距離を基準にした単位であり，定数kが１になるのは地球を基準にしているためだということに気がついた様子であった。この生徒に対する問いかけによって，生徒に定数kがほぼ１になることが重要ではなく，地球が基準となる単位を扱っているという本質に気づかせることができたと考えられる。

４．本実践から見出された課題

本章ではまず，本実践の授業展開を教科書の流れに沿って進められる従来の講義形式授業と比較する。

講義形式授業では，教師がケプラーの第三法則の関係式（$T^2=ka^3$）を与え，教科書に記載されている観測データ（a〔AU〕とT〔year〕）を用いると，定数kがほぼ1になることを表などを用いて確認する。この授業は，生徒がケプラーの第三法則に観測データが当てはまっているかどうかを“確認する”授業展開となる。

一方，本実践は観測データからケプラーの第三法則を導出する授業展開であった。その際，生徒は“プログラム順序や演算内容を自ら考え”，試行錯誤しながらケプラーの第三法則を導出することができた。また，生徒は単位の異なる観測データを同一プログラムによって分析・比較した。これにより，教科書の記述を吟味しながら，定数kがほぼ１になることが重要なのではなく，地球を基準にしていることが本質であることに気がついたと捉えられる。よって，本実践を通して生徒に物理の学習内容を深く考えさせることができたと判断できる。また，本実践の中で，数学科・情報科で扱われる決定係数R^2値や，地学分野で扱われる天文単位などを扱うことができた。これにより，教科・科目横断的な視点で学習内容を捉えることができたと考えられる。

本実践を通して，高等学校物理へのプログラミング導入に向けた困難が2つあると考えられる。

１つ目は，生徒が学習活動において情報端末を活用することの難しさである。多くの生徒はスマートフォンやタブレット端末を日常的に所持しており，日常生活における情報端末の使用は困難ではないと予想される。しかしながら，１人１台の情報端末環境はここ数年で急速に整備されてきたため，現在の高校生はこれまでの学習活動において情報端末を使用してきた経験が多いとは言えない。このことから，現在の高校生の情報端末使用は日常生活の域を出ないのではないかと推察される。本実践では，授業実践の導入後に Colab 操作マニュアルを共有した。これにより，導入では想定よりも多くの時間を費やしたものの，展開では生徒がそのマニュアルを参照しながらプログラム実行順序を効率的に考えることができたのではないかと考えられる。

２つ目は，テキスト言語におけるプログラム構造の視認性が低いことである。テキスト言語は数字や文字，記号のみでプログラムを記述するため，ビジュアル言語と比較してプログラム構造が複雑化しやすい。また，それぞれのテキスト言語特有の文法が存在することも多い。これらのことから，テキスト型言語のプログラムを生徒に記述させることは，テキスト言語の使用によほど慣れていない限り，困難であると考えられる。本実践では，プログラムを機能別にテキストファイル形式で分割し，生徒にどの順序でどのプログラムを実行させるかを考えさせた。これにより，テキスト言語である Python をビジュアル言語に近いプログラム構造で活用できるようになったと考えられる。また，プログラムを機能別に分割するこの方法は，小・中学校で多く用いられているビジュアル言語からテキスト言語に滑らかに移行する手立てとして有効なのではないかと考えている。

５．まとめ

平成29・30年の学習指導要領改訂によって求められているプログラミング教育の充実や，GIGA スクール構想の実現に向けた情報端末の整備状況を背景に，どの OS でも活用できるプログラミング教材を開発し

た。また，プログラミングと物理の親和性の高さを背景に，高等学校物理へプログラミングを導入し，生徒に物理の学習内容を深く考えさせる授業を検討した。

使用するプログラミング言語は価格や環境構築，複雑な計算処理対応の３観点から Python が最適であり，Python の環境構築をせずに Web ブラウザ上で扱うことのできるサービスとして，Colab を取り上げた。Colab を利用するにあたって Google アカウントを必要とするが，Colab には①端末を問わずに無料で活用できること，②プログラムの保存・共有が容易であること，③豊富な Python ライブラリを活用することができること，の3つのメリットがあった。

Colab を用いた本教材は，太陽系惑星の観測データからケプラーの第三法則を導き出すことができる教材である。本教材の特徴として，プログラムが6つのテキストファイルに分割されている点を述べた。これによって，生徒の思考活動は分割された6つのテキストファイル（プログラム）をどの順序で実行するかという点になる。最終的に，このプログラム作成を通して，論理的思考力の育成と物理の学習（ケプラーの第三法則）に帰結する。本教材を用いた授業実践により，生徒が一度無意識に受け入れた教科書の記述を吟味し，本質に気づくことができた。これは，教科書の流れに従った従来の講義形式授業と比較して，生徒が深く考える物理授業を実践することができたと考えている。

本実践を通して，高等学校物理へのプログラミング導入を見越した際，生徒の学習活動における情報端末使用の難しさと，テキスト言語におけるプログラム構造の視認性の低さの２つの困難が考えられた。本実践では，事前に操作マニュアルを作成・共有することにより，生徒にプログラミングを効率良く行わせることができたと考えられる。また，プログラムを機能別に分割することによって，生徒にビジュアル言語に近い構造でテキスト言語を活用させることができたと考えられる。Scratch 等のビジュアル言語はそのプログラム構造の視認性の高さから小・中学校のプログラミング教育で多く活用されている。一方で，Python 等のテキスト言語はこれまで教育現場への導入に障壁があった。しかしながら，プログラミング教材を工夫することによってテキスト言語を教育活動の中に取り入れ，生徒自らが考える授業を展開することが可能であると考えられる。

最後に，本研究は JSPS 科研費 19K03050 の助成を受け実施されたものである。

付記

本論文について，開示すべき利益相反関連事項はない。

引用・参考文献

Dreyer, J. L. E.(1953) A History of Astronomy from Thales to Kepler.Dover Publications.

池口良太・内山哲治（2009）LabVIEW を用いた波動シミュレーションの教材開発と授業実践．宮城教育大学情報処理センター年報，16:C1-C5.

加藤徳善（2004）直感的に操作できる物理シミュレーションソフトの開発．物理教育，52-3:259-261.

国立天文台（2021）理科年表プレミアム―惑星表―.https://www.rikanenpyo.jp/member/?module=Member&action=Contents&page=allASx11x0020_2021_1.html.（2022-12-21 Web閲覧）

文部科学省（2019）平成30年度告示 高等学校学習指導要領．東山書房．

文部科学省（2019）高等学校情報科「情報Ⅰ」教員研修用教材 第3章 コンピュータとプログラミング .https://www.mext.go.jp/content/20200722-mxt_jogai02-100013300_005.pdf.（2022-12-21 Web閲覧）

文部科学省（2020）教育の情報化に関する手引―追補版― 第3章 プログラミング教育の推進.https://www.mext.go.jp/content/20200608-mxt_jogai01-000003284_004.pdf.（2022-09-30 Web閲覧）

文部科学省（2021）端末利活用状況等の実態調査（令和3年7月末時点）（確定値）.https://www.mext.go.jp/content/20211125-mxt_shuukyo01-000009827_001.pdf.（2022-12-21 Web閲覧）

文部科学省（2022）高校学校における学習者用コンピュータの整備状況について（令和4年度見込み）.https://www.mext.go.jp/content/20220324-mxt_shuukyo01-000020467_001.pdf.(2022-12-21 Web閲覧)

奈良旬平（2020）プログラミングツール「Processing」を用いたアニメーション教材の開発と利用．東北物理教育，30:7-9.

能代谷賢治・内山哲治（2021）非インストール型Python実行サービス「Google Colaboratory」を用いた教材開発と授業実践．東北物理教育，31:9-14.

岡本恭介・安藤明伸（2020）ビジュアル型とテキスト型プログラミングにおける学習順序が教育的効果に与える影響．日本教育工学会論文誌，44:97-100.

竹ヶ原孝則・内山哲治（2015）LabVIEW を用いたヴィジュアル型シミュレーションの開発：投下速度直線運動と衝突運動．東北物理教育，25:20-23.

渡辺正雄（1991）ケプラーと世界の調和．共立出版．

Williams,D.R.(2021)Planetary Fact Sheet – Metric Units.https://nssdc.gsfc.nasa.gov/planetary/factsheet/index.html.（2022-12-21 Web閲覧）

山口智輝・内山哲治（2013）打撃シミュレーション教材を利用した物理の授業実践．宮城教育大学情報処理センター研究紀要，20:51-56.

山本義隆（2014）世界の見方の転換3 世界の一元化と天文学の改革. みすず書房.
本教材のダウンロードURL.https://drive.google.com/drive/folders/1YNNkFHG1A1kJA3iFrYpW3esnMy_CNf6g?usp=sharing（2023-03-09現在）

付録　本教材のプログラムコードおよび操作方法

本教材は太陽系惑星の観測データ（軌道長半径aと公転周期T）から，ケプラーの第三法則（$T^2=ka^3$ k:定数）をプログラミングによって見出す教材である。本教材で用いた太陽系惑星の観測データは2種類ある。1つ目の観測データ（data.txt）には，太陽系惑星の軌道長半径a〔AU:天文単位〕と公転周期T〔year〕が記述されている（表2）。この観測データには，理科年表プレミアム2021に記載されている惑星表のデータを用いた（国立天文台，2021）。2つ目の観測データ（data2.txt）には，太陽系惑星の軌道長半径a〔m〕と公転周期T〔day〕が記述されている（表3）。この観測データには，NASAがWeb上に公開している惑星表のデータを用いた（Williams, 2021）。

本教材のプログラムは，以下の表4に示す6つのテキストファイルに分割されており，表4に示す実行順序の通りに実行しなければ，Colab上でエラーが表示されるように設計されている。以下では，表2に示す観測データを用いたサンプルプログラム（Kepler.ipynb）を基にそれぞれのプログラムの説明を行う。

（ⅰ）では，数値計算を行うNumPyとグラフ描画等を行うmatplotlib，回帰分析等を行うscikit-learnの3種類のライブラリをColabに読み込ませる。ここでは，グラフを描画する機能としてmatplotlibのpyplotを使用し，モデル評価を行う機能としてscikit-learnのmetricsを使用する。

（ⅱ）では，太陽系惑星の観測データをColabに読み込ませ，軌道長半径をa，公転周期をTとする。ここでは，NumPyのテキストファイルを読み込む機能（loadtxt）を用いて，小数を扱うことのできるデータ（float型）として読み込んでいる。プログラムコード内のテキストファイル名を変更すると，data2.txtを使用することもできる。なお，サンプルプログラムコード内では観測データの“2行目”を指定する際に“1”，“3行目”を指定する際に“2”を用いている。これはPythonの場合，行列番号が0から始まること

表2　太陽系惑星の観測データ（国立天文台，2021）
1～3行目はPythonが読み取るデータ列を指す

惑星名	番号（1行目）	軌道長半径 a〔AU〕（2行目）	公転周期 T〔year〕（3行目）
水星	1	0.3871	0.24085
金星	2	0.7233	0.61520
地球	3	1.0000	1.00002
火星	4	1.5237	1.88085
木星	5	5.2026	11.8620
土星	6	9.5549	29.4572
天王星	7	19.2184	84.0205
海王星	8	30.1104	164.7701

表3　太陽系惑星の観測データ（Williams, 2021）
1～3行目はPythonが読み取るデータ列を指す

惑星名	番号（1行目）	軌道長半径 a〔m〕（2行目）	公転周期 T〔day〕（3行目）
水星	1	5.791×10^{10}	87.97
金星	2	1.082×10^{11}	224.70
地球	3	1.496×10^{11}	365.24
火星	4	2.279×10^{11}	686.98
木星	5	7.783×10^{11}	4332.59
土星	6	1.429×10^{12}	10759.22
天王星	7	2.875×10^{12}	30685.4
海王星	8	4.504×10^{12}	60189

表4　テキストファイル内のプログラム内容と実行順序

実行順序	ファイル名(.txt)	プログラム内容
（ⅰ）	import module	ライブラリの読み込み
（ⅱ）	load data	データの読み込み
（ⅲ）	process data	データの処理（演算）
（ⅳ）	fitting data	近似直線の算出
（ⅴ）	evaluation	近似直線のR^2値の算出
（ⅵ）	make graph	グラフの作成・表示

に依る。

（iii）では，a や T の値に演算を行い，a の演算結果を x，T の演算結果を y とする。Python で用いられる演算子は，加法が“+”，減法が“-”，乗法が“*”，除法が“/”であり，累乗は“**”で表される。ここでは，a を３乗した値を x，T を２乗した値を y とする例を取り上げる。

（iv）では，x と y の値に関する近似直線を算出・表示する。ここでは，NumPy の回帰分析を行う機能（polyfit）を用いて，近似直線の傾きおよび切片を最小二乗法によって算出している。表２の観測データを用いた場合，実行結果は以下のように表示される。

array(〔0.99450069, 0.1557849〕)

この表示は，近似直線の傾きが0.99450069，切片が0.1557849であることを示している。つまり，以下の関係式が成り立っている。

$$y = 0.9945x + 0.1558$$

$$(\ T^2 = 0.9945a^3 + 0.1558\ )$$

（v）では，fitting data.txt で算出した近似直線の R^2 値（決定係数）を算出する。ここでは，metrics の回帰モデルの予測精度を評価する機能（r2_score）を用いて，近似直線の評価を行う。表２の観測データを用いた場合，算出された R^2 値が以下のように表示される。

0.9999999994428102

（vi）では，データプロットと近似直線を表示する２次元グラフを作成・表示する。ここでは，pyplot を用いて横軸を x，縦軸を y とする２次元グラフを作成・表示している。data.txt を用いた場合，実行結果は図７のように表示される。

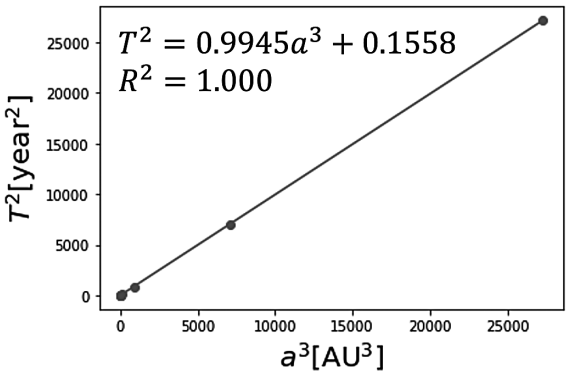

図７　T^2-a^3 グラフ（関係式は後から追加）

（iv）によって算出された近似直線の関係式を有効数字４桁で表すと，以下の式となる。

$$T^2 = 0.9945a^3 + 0.1558$$

この関係式の両辺を a^3 で割ると，以下の式となる。

$$\frac{T^2}{a^3} = 0.9945 + \frac{0.1558}{a^3}$$

ここで，右辺第２項の値はおよそ 10^{-11} や 10^{-12} 程度の大きさになるため，０に近似することができる。第２項の近似を行うと，以下の関係式が得られる。この関係式から，ケプラーの第三法則（$T^2=ka^3$）を導くことができる。

$$\frac{T^2}{a^3} \approx 0.9945$$