Jupyter Notebook を用いた数学教材におけるデザインの改善

神戸大学 大学院 人間発達環境学研究科　長坂耕作[1]

1　はじめに

近年では，高等学校「情報 I」におけるプログラミング教育の開始 [14] や内閣府の AI 戦略 2019[13] による数理データサイエンス教育の推進などもあり，中等数学教育においても，以前にも増して積極的にプログラミング環境を活用する試みが行われている。実際，神戸大学附属中等教育学校では,「データサイエンス I」の授業において，Google Colaboratory を利用して Python を用いた推測統計を取り扱う試み [12] が行われている。また，中等数学教育に限らず，研究領域に近い部分での高度な実験数学を Jupyter Notebook を実行基盤として実践する試み（横山ら [10, 11]）や，汎用の教材作成基盤として Jupyter Notebook を活用しようとする事例（北本 [4]）も報告されている。加えて,「情報 I」の教科書で用いられている主なプログラミング言語の 1 つが Python であるため [7]，Google Colaboratory や JupyterLab などの Jupyter Notebook を用いた Python 環境の活用は今後も広がっていくものと考えられる。

このような背景のもと，神戸大学国際人間科学部で開講している「数学科教育論 D」（資格免許科目，中等数学科の教科指導法）では，Jupyter Notebook を用いた教材作成を学習目標としている。表 1 に授業内容を抜粋しているが，Python 実行環境として JupyterLab を用い，Jupyter Notebook 上で数式処理（SymPy）と可視化（Bokeh）を中心に多面的な活用を目指している。この授業では，中等数学科を対象として，Jupyter Notebook を用いた教材作成を課題としている。授業開始当時は，Python 環境としての機能性に着目していたこともあり，履修者から提出された Jupyter Notebook のデザインに問題意識を持つことはなかった。しかしながら，数年間の授業経験を振り返ると，課題として提出された Jupyter Notebook の中にはデザインを主因として理解しづらいものも散見された。そこで，Jupyter Notebook を数学教材として活用する際のデザインの観点からの課題，即ち，教材としての質を左右しかねないデザイン上の課題を提示し，その解決策を提案する。

1.1　数学教材としてのデザイン上の課題

数学教材として Jupyter Notebook を活用する形態には概ね次の 4 つが考えられるが，本稿では，最後に述べる「学習者（生徒）が資料として用いる」形態について，その課題と解決策を提案する。

教授者（教師）が操作する

紙媒体の教科書や黒板に代わって動的に変化する様を学習者に提示するために教授者が操作するもの。主に，関数描画ソフトウェアや動的幾何ソフトウェアを用いて，連続的に変化する様子を可視化 [8] する実践例が多い。端末の普及に伴い学習者が操作するものに統合されつつある。

[1]E-mail: nagasaka@main.h.kobe-u.ac.jp

第 1 回目	数式処理ソフトウェアとインタラクティブな提示教材に触れる
第 2 回目	数式処理ソフトウェアの基本操作を身に付ける
第 3 回目	数式処理ソフトウェアで数学の問題を解く
第 4 回目	中等数学教育の単元と数式処理ソフトウェアの機能の関係
第 5 回目	資料作成のためのプログラミング入門 – 多項式の計算表示
第 6 回目	資料作成のためのプログラミング入門 – 条件を満たす分数
第 7 回目	インタラクティブなコンテンツの作成
第 8 回目	インタラクティブなグラフの教材
第 9 回目	数式処理ソフトウェアとインタラクティブな教材
第 10 回目	代数的性質に基づく，インタラクティブな初等幾何
第 11 回目	代数的性質に基づく，インタラクティブな初等幾何（その 2）
第 12 回目	数式処理ソフトウェアで可能な自習教材 – プリント作成
第 13 回目	インタラクティブなコンテンツの作成
第 14 回目	インタラクティブな二次曲線の教材
第 15 回目	数式処理ソフトウェアで可能な自習教材 – 設計
第 16 回目	数式処理ソフトウェアで可能な自習教材 – 実現

表 1: 数学科教育論 D のシラバスの授業計画より抜粋

教授者（教師）が資料作成に用いる

配布または提示することを目的としたグラフや表，例題や試験問題などを作成するもの。市販されている問題 DB の利用が拡大 [6] しており，この形態での利用は少ないと思われる。

学習者（生徒）が操作する

数学的な活動として学習者にスライダーや点などの操作を委ねるもの。特定の性質を満たす図形の条件を探すものなど，動的幾何ソフトウェアを用いた実践報告が多い [5, 3, 9]。Bokeh や Jupyter Widgets を用いて再現可能な事例も多い。

学習者（生徒）が資料として用いる

紙媒体の教科書やプリントに代替し，各時の主題導入部分から主な活動の全般（ないしは部分）の説明や図表等を示すもの。場合により上記 3 形態の全てを含み，デジタル教材として既存教材の代替を目指すもの（学習者用デジタル教科書など）。

基本的に，Jupyter Notebook はプログラミング開発を主とする統合開発環境であり，一般的な数学教材のデザインと異なる。図 1 は，Jupyter Notebook の Markdown セルの機能を活用して，適切な構造を付与したものであるが，図 2 や一般的な教科書や数学教材などと比べて，視覚的な面での工夫は見られない。これらの差は，Jupyter Notebook が教材作成ツールではなく，計算可能な文書を作成し共有するためのもの[2]という目的の違いが一因と考えられる。しかしながら，数学教育における教材としては，生徒が容易に理解した上で円滑に活用できるよう，本稿では視覚的な面での改善方法を提案する。

[2] “The Jupyter Notebook is the original web application for creating and sharing computational documents. It offers a simple, streamlined, document-centric experience.” （参考文献 [2] を参照のこと）

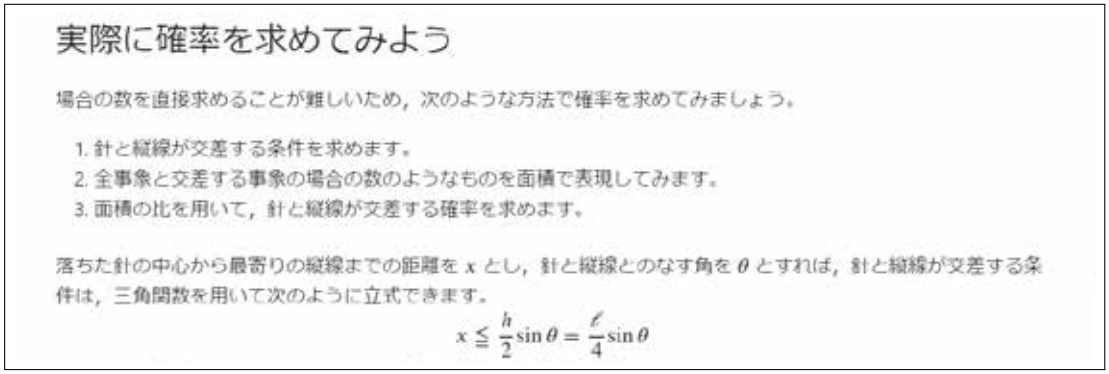

図 1: Jupyter Notebook の教材としてのデザイン（標準状態）

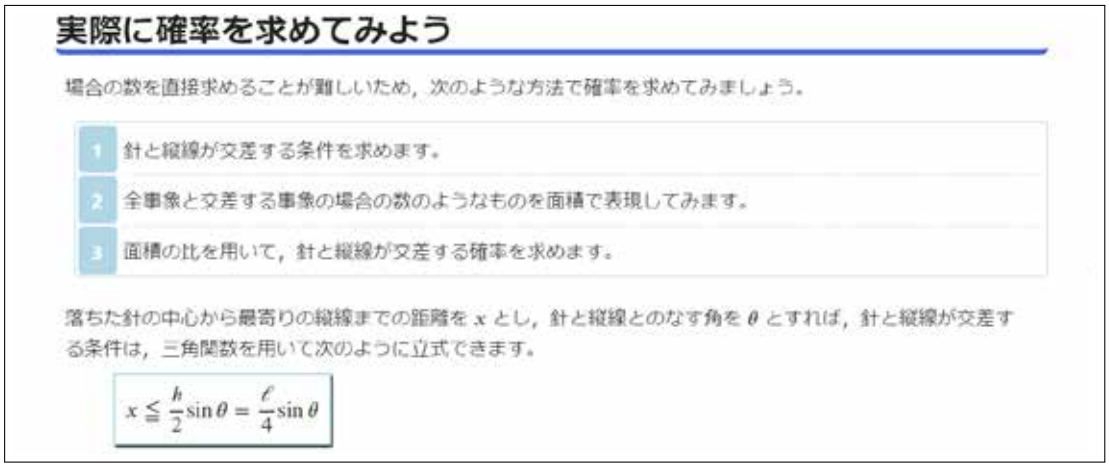

図 2: Jupyter Notebook の教材としてのデザイン（本稿方式）

2 JupyterLab のデザイン変更

2.1 標準の方法 – Theme の作成と変更 –

JupyterLab のデザインはテーマ（Theme）と呼ばれ，標準では「JupyterLab Light」と「JupyterLab Dark」のみがメニューから切り替え可能となっている。これ以外のテーマは，拡張機能として作成するか，公開されている拡張機能として作られたテーマをインストールすることで，利用可能となる。テーマでは，JupyterLab 自体のユーザーインターフェイス部分も含めて，様々な部分のデザインを操作することが可能である。しかしながら，本稿の目的としては教材部分のデザインが改善できればよく，ユーザーインターフェイス部分の改善については必ずしも必要ない。また，テーマでのデザイン改善には次のことを学習者側に強制することとなり，必ずしも意図したデザインが実現されるとは限らない。

- 拡張機能として指定したテーマをインストールすること
- Jupyter Notebook を開く際に意図したテーマを適用すること

このため，本稿では次節で導入する別の方法を採用し，教授者から学習者に対して教材として配布する Jupyter Notebook だけで意図したデザインを実現する方法を提案する。

2.2 提案する方法 – Jupyter Notebook 別のデザインの実現 –

2.2.1 Jupyter Notebook からのテーマの修正

テーマをインストールせずに，テーマで変更可能なデザイン（テーマを実現する拡張機能内で設定を行う CSS を規定する各種変数）を Jupyter Notebook 内から変更する方法が，Grout[1] により提案されている。この項では，その方法を簡単に紹介しておく。

1. 標準テーマで設定されるデザインを決定する CSS 変数（カスタムプロパティ）を定めるファイル（`variables.css`）を「https://github.com/jupyterlab/jupyterlab/blob/master/packages/theme-dark-extension/style/variables.css」から入手する。
2. 意図するデザインとなるようファイル（`variables.css`）内の CSS 変数を変更する。
3. ファイル（`variables.css`）を適用したい Jupyter Notebook が保存されているフォルダと同じフォルダに配置する。
4. Jupyter Notebook で次の命令を実行する（出力を消去しない限り再実行は不要）。

```
from IPython.display import display, HTML
with open('variables.css') as f:
    css = f.read().replace(';', ' !important;')
display(HTML('<style type="text/css">%s</style>CSS loaded.'%css))
```

2.2.2 Jupyter Notebook 別のデザイン手順

前項の方法では，標準テーマで設定されているデザインの修正は可能であるが，その範囲を超える変更が難しい。また，配布した Jupyter Notebook は，JupyterLab で開かれるとは限らず，以前の Jupyter 上で開かれる可能性もあり，JupyterLab の標準テーマに依存しない形でデザインを指定する方が望ましい。本稿が提案する方法では，Grout の方法で CSS を Jupyter Notebook に埋め込みつつも，対象とするものをテーマの CSS 変数ではなく，JupyterLab や Jupyter で使用される実際の CSS クラスとする。対象とする CSS クラスは，各種ブラウザの開発者ツールから調べることができる。また，同時に，開発者ツールから変更を行うことで，どのような修正を Jupyter Notebook 内に埋め込むべきかの確認も行える（図 3 は，実際の開発者ツールでクラスを調べ，修正を試している様子）。

つまり，次のような手順となる。

1. 適用する CSS を開発者ツールなどで調べ，ファイル（`textbook.css`）に保存する。
2. ファイル（`textbook.css`）を適用したい Jupyter Notebook が保存されているフォルダと同じフォルダに配置する。
3. Jupyter Notebook で次の命令を実行する（出力を消去しない限り再実行は不要）。

```
from IPython.display import display, HTML
with open('textbook.css') as f:
    css = f.read().replace(';', ' !important;')
display(HTML('<style type="text/css">%s</style>CSS loaded.'%css))
```

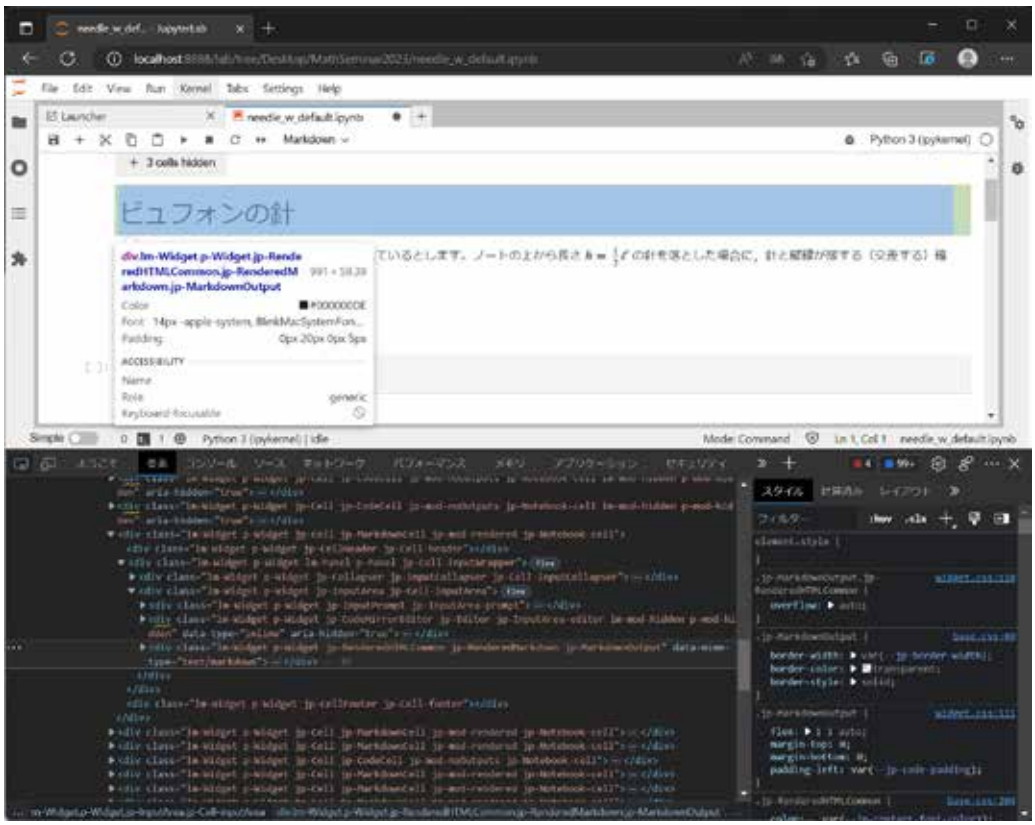

図 3: JupyterLab で開いた Jupyter Notebook の CSS の修正案を試している様子

本稿の例（図 2）では，主に次の CSS クラスを修正している。最後の 2 つを除き，カンマで区切られた左側が JupyterLab 向けで，右側が Jupyter 向けのものに対応する。それぞれ子孫セレクタを併用した指定となっており，最後の 2 つについては JupyterLab 向けのものとなる（Jupyter 向けにも記載する場合は，`code_cell input_area` と `code_cell output_subarea` などを用いる）。なお，中等数学教育を対象とした動的コンテンツを含む教材作成を念頭に置いているため，`Bokeh` による動的な操作が難しい Google Colaboratory は対象としていない。

`.jp-MarkdownCell, .text_cell`
Markdown セルに適用されるスタイル（編集内容も出力結果もどちらも対象）
`.jp-MarkdownOutput h1, .text_cell_render h1`
Markdown セルで「`#`」を用いて指定したレベル 1 の見出しの出力結果に適用されるスタイル
`.jp-MarkdownOutput h2, .text_cell_render h2`
Markdown セルで「`##`」を用いて指定したレベル 2 の見出しの出力結果に適用されるスタイル
`.jp-MarkdownOutput p, .text_cell_render p`
Markdown セルに記入した文章の出力結果に適用される基本的なスタイル

`.jp-MarkdownOutput .MathJax_Display, .text_cell_render .MathJax_Display`
Markdown セルに記入した MathJax の数式ブロック（「$$」で囲む形式のディスプレイスタイル部分）の出力結果に適用されるスタイル
`.jp-MarkdownOutput img, .text_cell_render img`
Markdown セルにアタッチメントとして埋込んだ画像の出力結果に適用されるスタイル
`.jp-MarkdownOutput blockquote, .text_cell_render blockquote`
Markdown セルに「>」を用いて指定した引用文の出力結果に適用されるスタイル
`.jp-CodeCell .jp-Editor`
Code セルの編集部分に適用されるスタイル
`.jp-CodeCell .jp-OutputArea-output`
Code セルの実行結果に適用されるスタイル

以上の方法により，Jupyter Notebook 別のデザインは実現可能となるが，教材配布上の問題が依然として二つ残る。今回の方法では，作り込んだ CSS を記載したファイルも共に配布する必要があり，配布ファイルが増えてしまうことが一つ。そして，Jupyter Notebook と CSS ファイルを共に配布できたとしても，Jupyter Notebook と対応する CSS ファイルが同一のフォルダに存在しなければならないことが一つである（Jupyter Notebook 内の出力を一斉消去する操作を行わなければ，再実行は不要であるが，必要となる可能性は否定できない）。この問題を解決する方法を次項で提案する。

2.2.3 デザイン（CSS ファイル）の Jupyter Notebook への埋込み

本項では，教材として利活用する Jupyter Notebook とは別に CSS ファイルも配布することは現実的ではないと考えられるため，Jupyter Notebook 単体でデザインを反映可能にする方法を提案する。これを実現するもっとも単純な方法は，必要となる CSS の内容を，別ファイルではなく Jupyter Notebook 内に直接記載することである。学習者が操作し編集する Jupyter Notebook 内に教材とは直接関係しない命令等を書くことになるが，仮に学習者が変更してしまっても，デザインが意図しない状態に変化するだけであるため許容範囲であろう。しかしながら，この方法には一つ欠点があり，図 4 のように（赤字部分はすべて CSS の記述），CSS 部分が Jupyter Notebook 内のかなりの範囲を占拠してしまう。以下では，この欠点を CSS の LZMA 圧縮と Base64 での符号化で改善した方法を提案する。

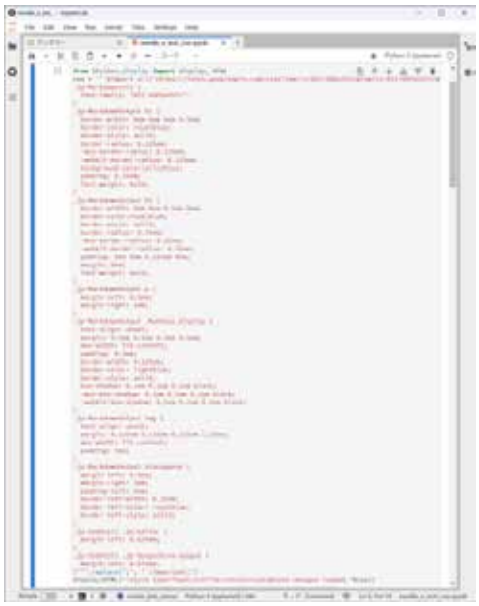

図 4: テキストのまま記載した場合

1. 教材として配布する Jupyter Notebook とは別の Jupyter Notebook（または，何らかの Python の実行環境）で，以下の 4 つの作業を行い，デザインに対応する CSS を LZMA 圧縮したものを Base64 で符号化したものを求める。

   (a) str 型で変数に CSS の内容を格納する（例えばヒアドキュメントで記載）。

   ```
   css = '''
   .jp-MarkdownCell, .text_cell {
   （中略）
     margin-left: 0.625em;
   }
   '''
   ```

   (b) 格納した内容を，bytes 型に変換する。例えば，変換は「encode」メソッドを用いることで,「css.encode('utf-8')」で実現可能（以下は実行結果の例）。

   ```
   b'\n.jp-MarkdownCell, .text_cell {\n（中略）: 0.625em;\n}\n'
   ```

   (c) さらに，その結果を LZMA により圧縮する。例えば，標準で含まれる lzma モジュールに含まれる「compress」メソッドを用いることで（import は別途），「lzma.compress(css.encode('utf-8'))」で実現可能（以下は実行結果の例）。

   ```
   b'\xfd7zXZ\x00\x00\x04\xe6\xd6\xb4F（中略）\x00\x00\x00\x04YZ'
   ```

   (d) 最後に，その圧縮後のものを Base64 で符号化し，扱いやすい文字列に直す。例えば，標準で含まれる base64 モジュールに含まれる「b64encode」メソッドを用いることで（import は別途），次の命令で実現可能（以下は実行結果の例）。「base64.b64encode(lzma.compress(css.encode('utf-8')))」。

   ```
   b'/Td6WFoAAATm1rRGAgAhARYAAAB0L+Wj4（中略）HEZ/sCAAAAAARZWg=='
   ```

2. Jupyter Notebook で次の命令を実行する（出力を消去しない限り再実行は不要）。ただし,「b'XXXXXXX'」の部分は (d) の出力（Base64 で符号化した文字列）である。

   ```
   import lzma
   import base64
   import IPython.display
   css = lzma.decompress(base64.b64decode(b'XXXXXXX'))
                       .decode('utf-8').replace(';', ' !important;')
   IPython.display.display(IPython.display.HTML(
                '<style type="text/css">%s</style>CSS loaded.'%css))
   ```

以上の方法で Jupyter Notebook に埋め込んだ結果が図 5 である。Jupyter Notebook に占めるデザインに関係するセルが簡潔になっていることが分かる。また，CSS を直接掲載する方法に比べて，この方法には次の利点があると考えられる。

- CSS 部分が Jupyter Notebook 内のかなりの範囲を占拠してしまうことを避けられる。
- 教材として適切に設定したデザインを学習者が意図せず変更することを避けられる。
- 誤って変更してもデザインは崩れず，標準デザインになるだけで済む（CSS の具体的な記述部分が，LZMA 圧縮後に Base64 で符号化している関係上，誤った変更では，復号化などに失敗し，結果として標準デザインに切り替わるだけとなる）。

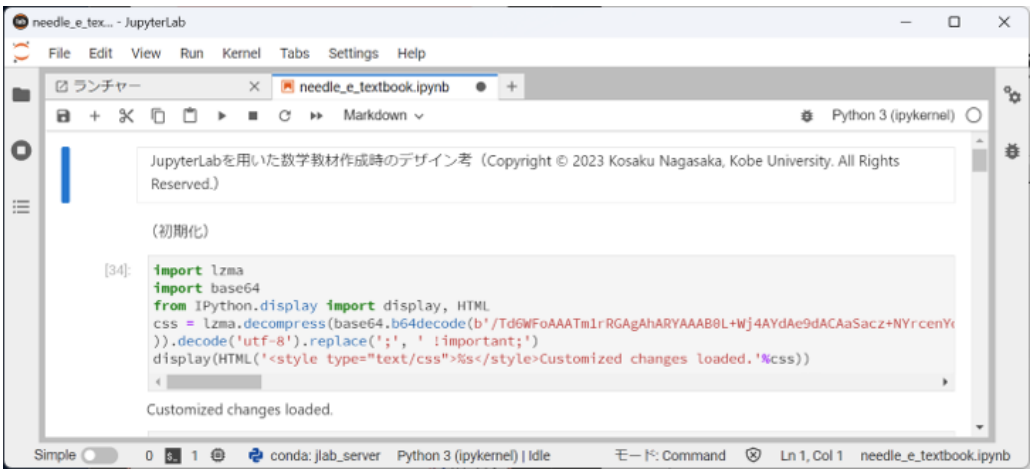

図 5: 符号化等を行って短くした文字列として記載した場合

3　まとめと今後の課題

本稿では，数学教育における教材として今後も利活用が進むと考えられる Jupyter Notebook において，プログラミング環境としての視覚的なデザインではなく，学習者にとり解りやすい，もしくは教科書等で頻出する慣例的なデザインを用いるべきであるとして，それを実現する方法を提案した。今回提案した内容により，学習者に教授者が意図した視覚的なデザインを容易かつ確実に届けることが可能となることは，数学教育における Jupyter Notebook の利活用の一層の推進に寄与するものと考える。

一方で，Jupyter Notebook を含むデジタルコンテンツ一般において，どのようなデザインが数学教育の教材として適しているかは明らかとなっていない。今後は，今回提案した方法を用いて設定可能なデザインのうち，数学教育で用いる Jupyter Notebook として最適なものの条件などを具体的に検討したい。

参考文献

[1] Jason Grout, JupyterLab Theme Customization (CustomizeJLabTheme.ipynb), https://gist.github.com/jasongrout/753216b2d3320b0abec6143d36f5d640, (2018), （参照: 2023/05/24）.

[2] Jupyter, https://jupyter.org/, (2014), （参照: 2023/05/25）.

[3] 恩田健介, 授業補助としての GeoGebra 教材の作成とその活用: 遠隔授業における実践報告書, 秀明大学紀要 **19**, (2022), 11–23.

[4] 北本卓也, Jupyter Notebook 上の教材作成について, 城西大学数学科数学教育紀要 **3**, (2022), 14–32.

[5] 北本卓也, 探究的な活動のための動的幾何ソフトウェアの活用について, 数式処理 **28(2)**, (2022), 35–51.

[6] 西浦孝治, 野澤武司, 高等学校における Studyaid D.B. の活用とその現状, 京都大学数理解析研究所講究録 **1951**, (2015), 162–166.

[7] 林良雄, 「情報 I」の教科書におけるプログラミングに関する用語の扱いについて, 秋田大学教育文化学部教育実践研究紀要 **45**, (2023), 155–162.

[8] 日野圭子, 比例の授業における数学的談話の構成: GeoGebra を通して教師が語ったこと, 宇都宮大学教育学部教育実践紀要 **2**, (2016), 145–154.

[9] 森岡正臣, 米川聡, 動的数学ソフト GeoGebra を利用した授業実践–中学校第 3 学年図形分野における授業づくりを通して–, 宮城教育大学紀要 **52**, (2018), 103–112.

[10] 横山重俊, 実験数学を教育から研究までやってみる, 京都大学数理解析研究所講究録 **2236**, (2022), 62–71.

[11] 横山重俊, 浜元信州, 長久勝, 谷沢智史, 藤原一毅, 政谷好伸, 竹房あつ子, 合田憲人, 実験数学を Jupyter Notebook でもっとやってみる, 京都大学数理解析研究所講究録 **2178**, (2021), 48–57.

[12] 神戸大学附属中等教育学校, 数学科の教育目標と公開授業の紹介, 研究紀要: 神戸大学附属中等 論集 **6** （別冊: 授業実践報告集）, (2022), 33–44.

[13] 内閣府 統合イノベーション戦略推進会議, AI 戦略 2019 〜人・産業・地域・政府全てに AI〜, https://www8.cao.go.jp/cstp/ai/index.html, (2019), （参照: 2023/05/24）.

[14] 文部科学省, 高等学校 学習指導要領（平成 30 年告示）, (2018).

付録

図 2 の Jupyter Notebook で修正している CSS の内容を参考までに以下に掲載する。

```
@import url('https://fonts.googleapis.com/css2?family=BIZ+UDGothic&\
             family=BIZ+UDPGothic&family=BIZ+UDPMincho&display=swap');
.jp-MarkdownCell, .text_cell {
  font-family: "BIZ UDPGothic";
}
.jp-MarkdownOutput h1, .text_cell_render h1 {
  border-width: 0em 0em 0em 0.5em;
  border-color: royalblue;
  border-style: solid;
  border-radius: 0.125em;
  -moz-border-radius: 0.125em;
  -webkit-border-radius: 0.125em;
  background-color:aliceblue;
  padding: 0.25em;
  font-weight: bold;
}
.jp-MarkdownOutput h2, .text_cell_render h2 {
  border-width: 0em 0em 0.2em 0em;
  border-color:royalblue;
  border-style: solid;
  border-radius: 0.25em;
  -moz-border-radius: 0.25em;
  -webkit-border-radius: 0.25em;
```

```
  padding: 0em 0em 0.125em 0em;
  margin: 0em;
  font-weight: bold;
}
.jp-MarkdownOutput p, .text_cell_render p {
  margin-left: 0.5em;
  margin-right: 1em;
}
.jp-MarkdownOutput > ol {
  counter-reset: olitem;
  list-style-type: none;
  border-width: 0.05em;
  border-style: solid;
  border-color: lightblue;
  border-radius: 0.25em;
  padding-top: 2px;
  margin-left: 1em;
}
.text_cell_render > ol {
  counter-reset: olitem;
  list-style-type: none;
  border-width: 0.05em;
  border-style: solid;
  border-color: lightblue;
  border-radius: 0.25em;
  padding-top: 2px;
  margin-left: 1em;
  padding-left: 2em;
  text-indent: -2em;
}
.jp-MarkdownOutput > ol > li, .text_cell_render > ol > li {
  border-bottom-width: 0.05em;
  border-bottom-color: lightblue;
  border-bottom-style: dashed;
  margin: 0.5em;
  padding-bottom: 0.5em;
}
.jp-MarkdownOutput>ol>li:last-of-type, .text_cell_render>ol>li:last-of-type {
  border-bottom: none;
  padding-bottom: 0;
}
.jp-MarkdownOutput > ol > li:before {
  counter-increment: olitem;
  content: counter(olitem);
  text-align: center;
  vertical-align: middle;
  padding-bottom: 0.5em;
  position: absolute;
  left: 1.525em;
  width: 1em;
  height: 1em;
  -moz-transform: translateY(-0.5em);
  -webkit-transform: translateY(-0.5em);
  transform: translateY(-0.5em);
  border-width: 0.5em;
  border-color: lightblue;
  border-style: solid;
  border-radius: 0.25em;
  color: white;
  font-weight: bold;
  background:lightblue;
}
.text_cell_render > ol > li:before {
  counter-increment: olitem;
  content: counter(olitem);
  text-align: center;
  vertical-align: middle;
```

```
  width: 1em;
  height: 1em;
  margin-right: 0.25em;
  -moz-transform: translateY(-0.5em);
  -webkit-transform: translateY(-0.5em);
  transform: translateY(-0.5em);
  border-width: 0.5em;
  border-color: lightblue;
  border-style: solid;
  border-radius: 0.25em;
  color: white;
  font-weight: bold;
  background:lightblue;
}
.jp-MarkdownOutput .MathJax_Display, .text_cell_render .MathJax_Display {
  text-align: unset;
  margin: 0.5em 0.5em 0.5em 2.5em;
  max-width: fit-content;
  padding: 0.5em;
  border-width: 0.125em;
  border-color: lightblue;
  border-style: solid;
  box-shadow: 0.1em 0.1em 0.1em black;
  -moz-box-shadow: 0.1em 0.1em 0.1em black;
  -webkit-box-shadow: 0.1em 0.1em 0.1em black;
}
.jp-MarkdownOutput img, .text_cell_render img {
  text-align: unset;
  margin: 0.125em 0.125em 0.125em 1.25em;
  max-width: fit-content;
  padding: 0em;
}
.jp-MarkdownOutput blockquote, .text_cell_render blockquote {
  margin-left: 0.5em;
  margin-right: 1em;
  padding-left: 0em;
  border-left-width: 0.25em;
  border-left-color: royalblue;
  border-left-style: solid;
}
.jp-CodeCell .jp-Editor {
  margin-left: 0.625em;
}
.jp-CodeCell .jp-OutputArea-output {
  margin-left: 0.625em;
}
.jp-collapseHeadingButton {
  min-height: 32px;
  min-width: 32px;
  background-color: transparent;
  background-size: 32px;
  background-repeat: no-repeat;
  background-position-x: center;
  background-position-y: 50%;
  border: none;
  background-image: url("data:image/png;base64（省略）,");
}
.jp-collapseHeadingButton:hover {
  background-color: aliceblue;;
}
.jp-collapseHeadingButton.jp-mod-collapsed {
  background-image: url("data:image/png;base64（省略）,");
}
```