

Markdown で \LaTeX する

2022 年 6 月 14 日

1 Markdown で \LaTeX する

Markdown は軽量のプログラミング言語である。数式をライブでレンダリングする MathJax, KaTeX を使えるところに目をつけて数式を書くために使ってみた。 \LaTeX との違いはコンパイルの概念がないことだろうか。とはいえ導入に一筋縄ではいかないことに気付いたのでメモしておく。

1.1 VSCode のユーザースニペット

初期設定では off になっているので忘れずに on にしておくこと。早速だが下の表を見てほしい。

	latex	md	.tex	.md	“latex”
.	O	O	×	O	×
/	O	O	×	O	×
___ と **	O	×	O	×	O
@	×	×	O	×	×
\	×	×	O	O	×

これは VSCode のユーザースニペットで一文字でスニペットを出してくれるか出してくれないかを表す表である。先頭から jupyter notebook において latex は青と黄色にハイライトされている時、md はオレンジの時、.tex, .md はそれぞれ拡張子 .tex, .md のファイルで開いた時、“latex” は jupyter の言語設定を “latex” に変えたものである。この表は相当錯綜しているが、なんとなく Markdown では \LaTeX でいうところの @ と \ の代わりに . と / が使えそうだとわかるはずだ。下は上から latex と md, “latex”, .tex, .md の時をとった写真である。マークダウン中で数式がオレンジになっているところがマークダウン、青や黄色が latex の扱いらしい。

```
1 | $\sin x$ $\sin x$
2
3 $$
4 \begin{pmatrix}
5 a_{11}& a_{12} \\
6 a_{21}& a_{22}
7 \end{pmatrix}
8 $$
9
10 |
```

Markdown

```
1 | $\sin x$ $\sin x$
2
3 $$
4 \begin{pmatrix}
5 a_{11}& a_{12} \\
6 a_{21}& a_{22}
7 \end{pmatrix}
8 $$
```

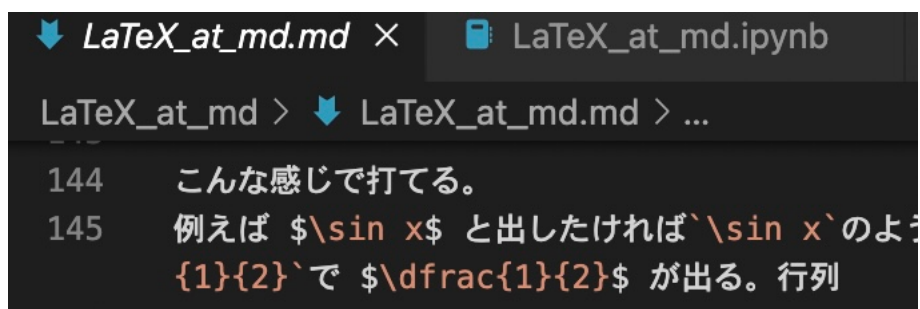
LaTeX

TeX LaTeX_at_md.tex ×

LaTeX

~/Documents/VSCode用
LaTeX_at_md.tex

```
546
547 \[
548 \begin{pmatrix}
549 a& b \\
550 c& d
551 \end{pmatrix}
552 \]
553
554 を出したければ
```



実際打ってみるとこんな感じ

[動画](#)

1.2 ユーザーズニペットの定義の仕方

1.2.1 I^AT_EX からの輸入

基本的には I^AT_EX のように数式を打つのだから全く同じように打ちたいと考える。どうやって打つかがきもになる。LaTeX Workshop に準拠したいと考えるなら PC 上の/Users/home/.vscode/extensions/james-yu.latex-workshop-8.27.2/data/at-suggestions.json から @ で始まるスニペットを輸入するのがおすすめ。とはいえ元のコードを書き換えるわけにはいかないからコピペするのが最も安全だろう。/Users/home/.vscode/extensions/james-yu.latex-workshop-8.27.2/snippets/latex.json にもその他の役に立つスニペット（フォントに関することは Markdown では使わない）があるのでこちらもコピペがおすすめ。

1.2.2 変換

輸入したスニペットはそのままでは使えない。上の表にある仕様のせいである。回避するには例えば cmd+shift+L を押すなどして at-suggestions 内の全コードの @ を . または / に変えてしまうのがいいだろう。ついでに例えば \alpha のコードをちょっといじって \alpha にしてしまう（空白を入れる）のもおすすめ。こうするとスニペットで \alpha を出した後に空白を打たなくて済む。

1.2.3 割り当て

コピーした上でどの記号を何に割り当てるのは結構重要である。ちなみに自分は主要なものだと下のようにしている。

```
~/Library/Application Support/Code/User/snippets/latexmd.code-snippets
"inline_math2": {
  "prefix": "/",
  "scope": "latex,markdown",
  "body": " $$1$ $0"
},
"\\left( \\right)2": {
```

```

    "prefix": "..",
    "scope": "latex,markdown",
    "body": "\\left($1\\right)$0"
  },
  "\\left| \\right|2": {
    "prefix": ".|",
    "scope": "latex,markdown",
    "body": "\\left|$1\\right|$0"
  },
  "\\left{ \\right}2": {
    "prefix": ".{",
    "scope": "latex,markdown",
    "body": "\\left\\{$1\\right\\}$0"
  },
  "\\left[ \\right]2": {
    "prefix": ".[",
    "scope": "latex,markdown",
    "body": "\\left[$1\\right]$0"
  },
  " 別行立て数式 2": {
    "prefix": "ma",
    "body": [
      "",
      "$$",
      "${1}",
      "$$",
      "",
      "${0}"
    ]
  },
  "and2": {
    "prefix": ".",
    "scope": "latex,markdown",
    "body": "& "
  },
  "dfrac": {
    "prefix": "//",
    "scope": "latex,markdown",
    "body": "\\dfrac{$1}{$2}$0"
  },

```

```

"inline_code": {
  "prefix": ".,",
  "body": "~$1~$0"
},

```

ポイントは `scope` を `"latex,markdown"` にしておくこととスニペットファイルをグローバルにしておくことである。命令が既にあるものと重なるものは必ず 2 をつけて表した。 `inline_math2` は挙動が LaTeX Workshop の `inline_math` と近くなるようにしておいた。(向こうでは `\` を打つところをこっちでは `/` を打っている。前後に空白を空けている。) `left right` 系のコマンドはあまり変えなかったが `.|` を付け足したのと元の `@` (がちちょっと打ちにくいので `..` に変えておいた。別行立ての数式は普通に `math` のアルファベット。 `inline_code` は打ちやすい位置にした。

1.3 Jupyter Notebook を使う

自分は Markdown を Jupyter Notebook で打っている。ただの `.md` ファイルで打たないのは一定の利点があるためである。

1.3.1 セルの分割

Jupyter Notebook は本来 Python 等のコードを書くためのものであるからセルに分割されている。つまり途中セーブのようなことができて精神衛生的にいい。これは L^AT_EX に対しても言えることである。

1.3.2 コードの編入が楽

ソースコードを入れるときに一回 Python のセルに投げ込んでその言語に変えればソースコードのコードを見なくて済む。

1.3.3 jupyter-nbconvert が優秀

`jupyter-nbconvert` は Python を入れたときにデフォルトで一緒についてきたコマンドである。ターミナルで以下のコマンドを実行するだけで変換してくれる。例えば `html` にしたければ下のようにする。(`pdf,latex,markdown` 等にしたければそれぞれ帰れば勝手になる)

```
[ ]: jupyter-nbconvert --to html [変換したいファイルのパス]
```

この優秀なところは結構質の良い `html` ファイル、`tex` ファイルに変換してくれることである。特に `tex` ファイルへの変換の優秀さは `pandoc` のコマンドに比べて特筆すべきものがある。(なぜか Cloud LaTeX でもうまく `md` を変換してくれない) 要は `md` から `pdf` にするよりも `ipynb` から `jupyter-nbconvert` で `pdf` にする方が精度が高いというわけである。なお `ipynb` から変換したからといってセル毎に分割されて小さく組版されるなどということはない。

1.4 実際に数式を打ってみる

こんな感じで打てる。例えば $\sin x$ と出したければ `\sin x` のように打てば良い。分数を出したければ `\dfrac{1}{2}` で $\frac{1}{2}$ が出る。行列

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix}$$

を出したければ

```
$$
\begin{pmatrix} a & b \\ c & d \end{pmatrix}
$$
```

のようにすれば良い。マークダウンセルとしてコメントアウトしたければデフォルト?では `cmd+`. または `<!-- -->` で囲むことによって可能。

```
<!-- あああ コメントです いい -->
```

数式中で LATEX に従ってコメントすることもできる。例) 入力

```
$$
\sin x + \cos x
$$

$$
<!-- \sin x -->
$$
```

出力

$$\sin x +$$

```
<!-- sin x -->
```

であるので数式中で `<!-- -->` を使うのはグレーかもしれない。

1.5 その他

地味な設定ではあるが LaTeX Workshop には環境やコマンドで囲んでくれるコマンドがある。

`md_cmd_env`

これを使えるようにするには `keybindings.json` の `when` の箇所をなくす、または他の言語で反映されたくない人は `"editorTextFocus && editorLangId == 'latex'"` に `markdown` を付け加えておくが良い。本家 \LaTeX とほぼ同じような環境で数式を入力できるようになる。

$$\mathbb{R} \times$$

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix}$$

[]: