TEX/LATEX ことはじめ

cuttlefish_math

2021 年 8 月 9 日

目次

第 1 章　はじめに

TEX/LATEX は理工系で広く用いられているシステムであり，特に数学に関わる分野では，文書作成のデファクトスタンダードと言える．しかしながら，その全容について把握するのは容易ではない．

たとえば，TEX/LATEX により次の数式を記述したいとする．

$$f(t) \sim \frac{a_0}{2} + \sum_{n=1}^{\infty} (a_n \cos nt + b_n \sin nt)$$

あなたは「シグマ TeX」などと検索することで，概ね次のようなコードを書けばよいことが分かるだろう．

```
\[
  f(t) \sim \frac{a_0}{2} + \sum_{n=1}^\infty (a_n\cos nt + b_n\sin nt)
\]
```

しかし，こうして課題のレポートをなんとか作れるようになったとしても，「TEX/LATEX とは」と問われると答えに詰まるのではないだろうか．

本稿は「TEX/LATEX を理解して，自由に文書を作成するためのチュートリアル」となるべく書かれている．特に，理工系学生がよく利用するであろう「表作成」と「グラフ作成」に重きをおき，サンプルコードを多く載せることで，レポートを作るときに役立つようになっている．本稿が TEX/LATEX について，ブラックボックスとしてではない理解を得る一助になれば幸いである．

第 2 章　コンピュータとマルチメディア

2.1　文字コード

コンピュータにおいて，テキストを記録するには，テキストを数値の列に変換しなければならない．この変換方式のことを**文字コード**という．文字コードは複数あり，それぞれ利用できる文字や変換方式などが異なる．

たとえば，最新の Windows 10 にあるメモ帳でテキストを保存すると，UTF-8 という方式で文字列が数値の列に変換される．この他にも，ソフトウェアに応じて ASCII，Shift_JIS など，さまざまな文字コードが利用されている．

図 2.1: Windows 10 のメモ帳

現在では **Unicode** という文字コードの規格が主流になりつつある．先述の UTF-8 も，この Unicode において定義されている．この節では文字コードの代表例として，Unicode について解説する．なお，以下に示す用語は文字コードによっては異なる名前で呼ばれることもあるし，対応する概念が無いこともある．

文字コードが扱える文字の全体集合を**文字集合**という．文字集合の各要素には固有の非負整数が割り当てられており，その対応関係を Unicode では**符号化文字集合**と呼ぶ[1)]．

符号化文字集合（の像）は，集合 $S = \{0_{(16)}, \ldots, 10\mathrm{FFFF}_{(16)}\}$ の真部分集合である．S を**符号空間**といい，S の要素を**符号位置**という．符号位置は，十六進数の先頭に「U+」を付けて「U+304B」のようにして表される．

符号化文字集合は，数値を並べたときに区切れ目がただ 1 通りに定まるかどうかを考慮していない．そこで，あとから区切れ目が分かるように，符号化文字集合（の像）に属する数値を整数の組[2)]（**符号単位**）へと変換する規則が必要になる．この規則を**文字符号化形式**といい，UTF-8，UTF-16，UTF-32 の 3 つが定義されている．

$$\text{か} \xrightarrow{\text{符号位置に変換}} \text{U+304B} \xrightarrow{\text{符号単位に変換}} \texttt{E3 81 8B}$$

図 2.2: 文字の符号単位への変換

表 2.1 に Unicode の符号化文字集合を一部抜粋して示す．

例として「M 系列」という文字列について考えよう．「M 系列」の各文字を Unicode の符号位置に置き換えると次のようになる．

---

1)　対応関係なのに集合？と思うのももっともだが，公式資料には確かに「a mapping from an abstract character repertoire to a set of nonnegative integers」[6] とある．

2)　正確には 1 バイトの組である．

表 2.1: 符号化文字集合の一部

U+	0	1	2	3	4	5	6
0020		!	"	#	$	%	&
0030	0	1	2	3	4	5	6
0040	@	A	B	C	D	E	F

```
U+004D U+7CFB U+5217
```

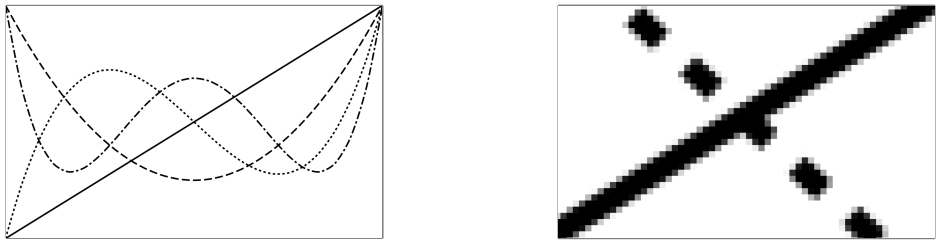
そして，これらを UTF-8 で符号単位の列に変換すると次のようになる[3)]．「4D」が U+004D，「E7 B3 BB」が U+7CFB，「E5 88 97」が U+5217 にそれぞれ対応する．

```
4D E7 B3 BB E5 88 97
```

2.2 ラスタ形式とベクタ形式

コンピュータで画像を記録する方法は，**ラスタ形式**と**ベクタ形式**の 2 つに大別される．ラスタ形式は，画像を単色の要素（**画素**）を集めたものとして記録する方法である．これに対し，ベクタ形式は画像を図形の集まりとして記録する方法である．

ベクタ形式に比べ，ラスタ形式は拡大・縮小の影響を受けやすい．ラスタ形式の画像を拡大したときの様子を図 2.3 に示す．

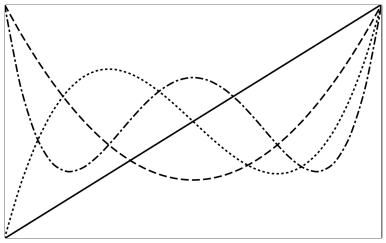


(a) 画像全体　　(b) 中央付近を拡大

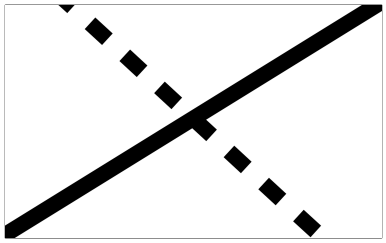
図 2.3: ラスタ形式の画像を拡大したときの様子

これに対し，ベクタ形式の画像は拡大しても図 2.3 のようにならない．これは，ベクタ形式では拡大・縮小に応じて図形を描画しなおせるからである．ベクタ形式の画像を拡大したときの様子を図 2.4 に示す．

3) 厳密には，UTF-16 と UTF-32 では符号単位の列をバイト列に変換する方法（**文字符号化スキーム**）も問題になる．UTF-8 では，符号単位を並べたものをそのままバイト列とする．



(a) 画像全体



(b) 中央付近を拡大

図 2.4: ベクタ形式の画像を拡大したときの様子

2.3 補遺

2.3.1 文字化けの仕組み

多くの場合，文字化けはテキストの文字コードをソフトウェアが誤判定したときに発生する．たとえば，図 2.5 では UTF-8 で作成したテキストを Shift_JIS で開いてしまっている．

文字化けを直すには，適切な文字コードでテキストを開きなおせばよい．Visual Studio Code[4)]などのテキストエディタは，ユーザが文字コードを選択しなおす機能を備えている．

図 2.5: 文字化けしたテキスト

4） https://code.visualstudio.com

第 3 章　TEX/LATEX とは

3.1　TEX とは

TEX とは Donald Knuth により開発された組版システムである．

3.2　TEX 処理系

upTEX は内部コードを Unicode にした pTEX の拡張である．

3.3　LATEX とは

LATEX とは Leslie Lamport により開発された TEX のマクロである．

3.4　補遺

3.4.1　ε-TEX について

実は，upLATEX を起動したときに実行されるのは，元々の upTEX とは少し異なる TEX 処理系である．

コマンドラインで `uplatex` を実行すると，次のメッセージが出力される．

```
This is e-upTeX, Version 3.141592653-p3.9.0-u1.27-210218-2.6 (utf8.uptex) (TeX
  Live 2021/W32TeX) (preloaded format=uplatex)
 restricted \write18 enabled.
**
```

本書では以降，ε-TEX 拡張の有無についてはいちいち断らないことにする．すなわち，upTEX と言ったら ε-upTEX を指すものとする．

第 4 章　グラフの作成

本稿では，次の 3 つの方法を紹介する．

1. PGFPlots による方法
2. GNUPlot による方法
3. TikZ の datavisualization ライブラリによる方法

4.1　datavisualization

TikZ

4.2　GNUPlot による方法

あああ[1)]．

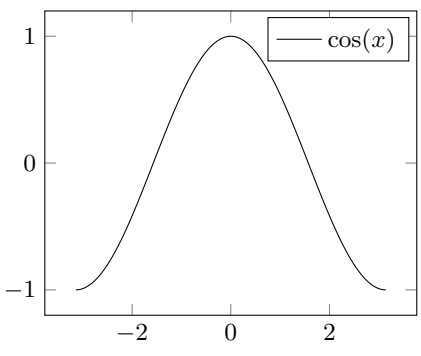
```
\begin{tikzpicture}
  \begin{axis}[width=7cm]
    \addplot [smooth,samples=100,domain=-pi:pi] {cos(deg(x))};
    \addlegendentry{$\cos(x)$};
  \end{axis}
\end{tikzpicture}
```

```
\input{gnuplot.tex}
```

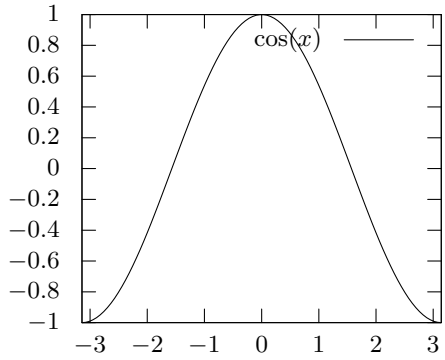
```
gnuplot> set terminal lua tikz size 7cm,5cm
gnuplot> set output "gnuplot.tex"
gnuplot> set xrange [-pi:pi]
gnuplot> plot cos(x) linetype rgb "black" title '$\cos(x)$'
gnuplot> exit
```

```
\begin{tikzpicture}
  \begin{axis}[width=7cm]
    \addplot [smooth,samples=100,mark=none,domain=-pi:pi] function [id=cos] {
      cos(x)};
```

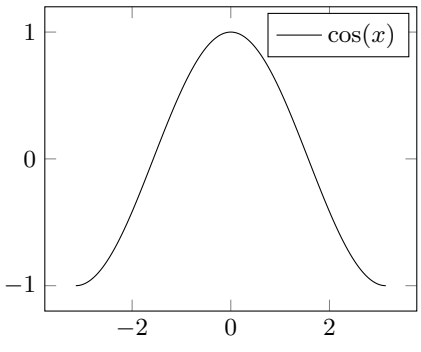
1)　`set term tikz` としている資料もあるが，これは `set terminal lua tikz` の省略形である．



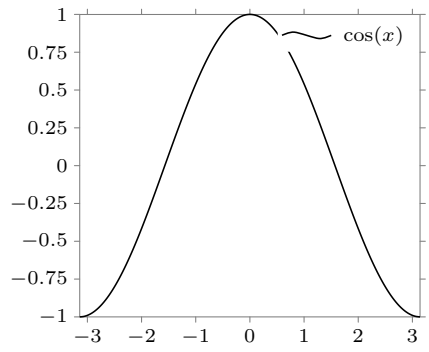
PGFPlots



gnuplot-lua-tikz



GNUPlot on PGFPlots



datavisualization

図 4.1: グラフ描画の比較

```
    \addlegendentry{$\cos(x)$};
  \end{axis}
\end{tikzpicture}
```

```
\begin{tikzpicture}
  \datavisualization [
    scientific axes,
    visualize as smooth line/.list={cosine},
    x axis={length=5cm},
    y axis={length=4cm},
    legend=north east inside,
    cosine={label in legend={text={$\cos(x)$}}}]
  data [set=cosine, format=function] {
    var x : interval [-pi:pi] samples 100;
    func y = cos(deg(\value x));
  };
\end{tikzpicture}
```

4.3 その他のグラフ

3D グラフなども同様の要領で作成できる．以下に PGFPlots を利用した例を示す．

```
\begin{tikzpicture}
  \begin{axis}[width=7cm,colormap/blackwhite]
    \addplot3 [surf,miter limit=1,samples=30,domain=-2:2] {exp(-(x^2+y^2))};
  \end{axis}
\end{tikzpicture}
```

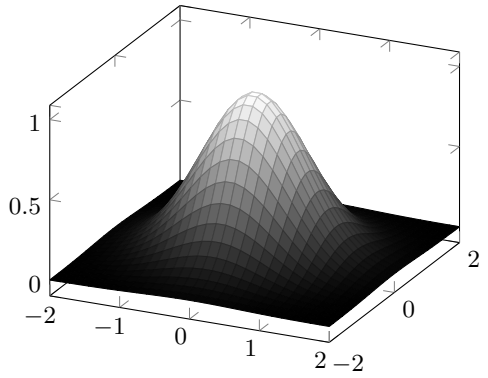


図 4.2: $e^{-(x^2+y^2)}$ のグラフ

4.4 補遺

4.4.1 グラフ描画の精度

PGFPlots の Revision 1.18 (2021/05/08) p.54 を引用する．

> **About the precision and number range:** Starting with version 1.2, `\addplot expression` uses a floating point unit. The FPU provides the full data range of scientific computing with a relative precision between 10^{-4} and 10^{-6}. The `/pgf/fpu` key provides some more details.
>
> Note that pgfplots makes use of lualatex's features: if you use lualatex instead of pdflatex, pgfplots will use lua's math engine which is both faster and more accurate (`compat=1.12` or higher).

つまり，処理系が LuaLaTeX であり，`compat` に 1.12 以上のバージョンが指定されていれば，PGFPlots は Lua の数学エンジンを利用して計算を行う．これは TeX 上の実装より高速かつ正確である．

第 5 章　フォント

この節では処理系を LuaTEX に限定する．これは，LuaTEX が日本語組版に対応しており，かつフォントを簡単に変えられる処理系だからである．

LuaTEX において，和文フォントの変更は luatexja-fontspec パッケージにより行う．たとえば，この文書ではプリアンブルに次のように書くことで，ソースコードのフォントを白源[1)]に変更している．

```
\setmonofont{HackGen Regular}[BoldFont={HackGen Bold}]
```

1） https://github.com/yuru7/HackGen

第 6 章　補遺

6.1　パッケージの相性問題

1. cleveref は amsmath の後に読み込まれなければならない
2. hyperref は xcolor の後に読み込まれると xcolor の色指定を利用できる
3. geometry は hyperref の後に読み込まれなければならない（あるいは geometry に hyperref オプションを渡す）
4. hyperref を読み込んだ場合，nameref は `\begin{document}` の直前に読み込まれる

これを見ると分かるが，特に hyperref に関するものが多い．

hyperref

> The environments equation and eqnarray are not supported too well. For example, there might be spacing problems (eqnarray isn't recommended anyway, see CTAN:info/l2tabu/, the situation for equation is unclear, because nobody is interested in investigating). Consider using the environments that package amsmath provide, e.g. gather for equation. The environment equation can even redefined to use gather:

```
\usepackage{amsmath}
\let\equation\gather
\let\endequation\endgather
```

参考文献

[1] Karl Berry, Torsten Martinsen, and Stephen Gilmore. LaTeX 2ε: An unofficial reference manual. https://latexref.xyz, 2021. Accessed: 2021-07-17.

[2] Karl Berry, Manuel Pégourié-Gonnard, and Jim Hefferon. A first set of LaTeX resources. https://ctan.org/pkg/latex-doc-ptr?lang=en, 2021. Accessed: 2021-07-17.

[3] Unicode Consortium. Unicode 13.0.0. https://unicode.org/versions/Unicode13.0.0/, 2020. Accessed: 2021-07-26.

[4] Tobias Oetiker, Hubert Partl, Irene Hyna, and Elisabeth Schlegl. The not so short introduction to LaTeX 2ε. https://ctan.org/tex-archive/info/lshort/english/?lang=en, 2021. Accessed: 2021-07-17.

[5] Mark Trettin and Jürgen Fenn. An essential guide to LaTeX 2ε usage. https://ctan.org/

tex-archive/info/l2tabu/english?lang=en, 2009. Accessed: 2021-07-17.

[6] Ken Whistler, Mark Davis, and Asmus Freytag. Unicode technical report #17: Unicode character encoding model. https://www.unicode.org/reports/tr17/tr17-7.html, 2008. Accessed: 2021-08-07.

索引