

TEX/LATEX

ことはじめ

ver.0.1.0

calamari_dev



目次

第 1 章 コンピュータとマルチメディア	4
1.1 文字コード	4
1.2 ラスタ形式とベクタ形式	5
1.3 可逆圧縮と非可逆圧縮	6
1.4 ライセンス	7
1.5 補遺	7
1.5.1 文字化けの仕組み	7
1.5.2 ラスタ形式と拡大・縮小	7
第 2 章 T_EX/L^AT_EX とは	9
2.1 T _E X と L ^A T _E X	9
2.2 T _E X を利用する方法	9
2.3 T _E X と和文	10
2.4 ドキュメントクラス	10
2.5 パッケージ	11
2.6 補遺	12
2.6.1 Texdoc	13
2.6.2 文書作成を支援するソフトウェア	13
2.6.3 ϵ -T _E X	13
2.6.4 他の命令の引数にできない命令	13
第 3 章 LuaL^AT_EX による実例	15
3.1 和文のドキュメントクラス	15
3.2 文献管理と索引作成	17
3.3 LuaL ^A T _E X によるフォント設定	17
3.4 グラフの作成	17

はじめに

本書の目的は「The Not So Short Introduction to L^AT_EX 2_ε」[6] に、L^AT_EX 2_ε と直接は関わらないが重要な知識と、和文の文書作成に関する事項を補完することである。特に本書の章 3 を読むにあたっては、[6] を傍らに置いて欲しい。

また、理工系学生がよく利用するであろう「グラフ作成」の方法を補い、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役立つようにしている。

本書の最新版は GitHub (<https://github.com/calamari-dev/texlatex>) で配布されている。GitHub には本書の L^AT_EX ソースも置いてある。また、この版 (ver.0.1.0) は CC BY-NC-ND 4.0 に基づいて配布されている。

2021 年 8 月 24 日

calamari_dev

第 1 章 コンピュータとマルチメディア

この章では、コンピュータで文字・画像を扱うにあたっての基礎知識を解説する。また、他者が製作した文書・画像を利用するときに重要となる、ライセンスについても説明する。

1.1 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。

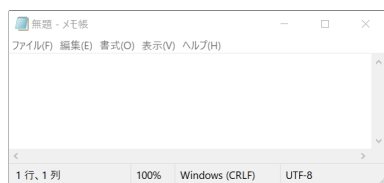


図 1.1: Windows 10 のメモ帳

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。

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と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 $S = \{0_{(16)}, \dots, 10FFFF_{(16)}\}$ 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。

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

図 1.2: 文字の符号単位への変換

表 1.1 に Unicode の符号化文字集合を一部抜粋して示す。

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」[8] とある。
2) 正確には 1 バイトの組である。

表 1.1: 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

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。

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。

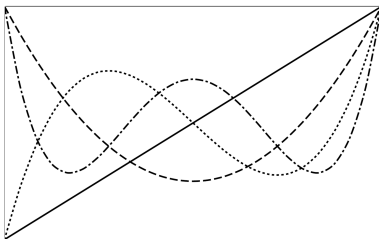
4D E7 B3 BB E5 88 97

1.2 ラスタ形式とベクタ形式

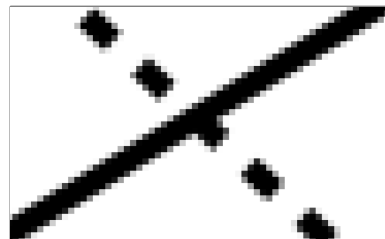
コンピュータで画像を記録する方法は、**ラスタ形式**と**ベクタ形式**の2つに大別される。ラスタ形式は、画像を単色の要素（**画素**）を集めたものとして記録する方法である。これに対し、ベクタ形式は画像を図形の集まりとして記録する方法である。

例を挙げると、**JPEG**、**PNG** はラスタ形式である。また、ベクタ形式である **SVG** は、Web ページのデザインにしばしば利用されている。

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 1.3 に示す（ただし、ラスタ形式の画像を拡大・縮小した結果は利用したソフトウェアに依存する。詳しくは小節 1.5.2 を参照のこと）。



(a) 画像全体

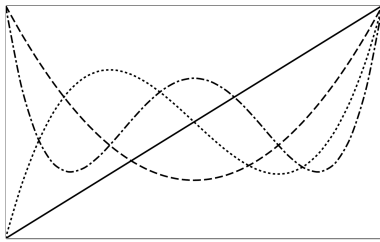


(b) 中央付近を拡大

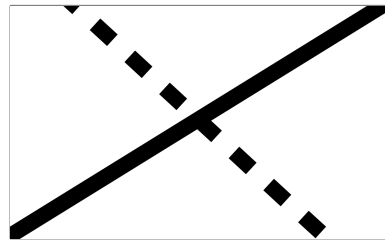
図 1.3: ラスタ形式の画像を拡大したときの様子

3) 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。

これに対し、ベクタ形式の画像は拡大しても図 1.3 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 1.4 に示す。



(a) 画像全体



(b) 中央付近を拡大

図 1.4: ベクタ形式の画像を拡大したときの様子

1.3 可逆圧縮と非可逆圧縮

この節では、ラスタ形式の画像について扱う。多くの場合、ラスタ形式の画像ファイルは、データサイズを削減するために圧縮されている。圧縮した後のデータから元のデータを復元できる圧縮方式を**可逆圧縮**、復元できない圧縮方式を**非可逆圧縮**という。

たとえば、PNG は可逆圧縮、JPEG は非可逆圧縮である⁴⁾。非可逆圧縮のほうがデータサイズが小さくなる傾向にあるが、画像の性質にもよるので一概には言えない。大雑把に言うと、PNG は線画、JPEG は写真に強い傾向にある。

非可逆圧縮を利用するときは、圧縮するたびに画像が劣化することに注意しなければならない。また、データサイズを削減するために品質をあまりに低く設定すると、画像が非常に粗くなることがある。図 1.5 に、JPEG の品質（0 以上 100 以下の整数を設定できる）を 10 に設定して圧縮したときの様子を示す。



(a) 元画像



(b) 圧縮した画像

図 1.5: 画像の過剰な圧縮に伴う劣化

4) JPEG の規格は可逆圧縮にも対応している。しかし、多くの場合 JPEG は非可逆圧縮に利用されるので、本書では JPEG は非可逆圧縮であると見なす。

1.4 ライセンス

通常、製作者は自身の製作物に関して知的財産権を持つ。したがって、他者の製作物を利用するとき、普通は製作者に対して許可をとる必要がある⁵⁾。しかし、製作物の**ライセンス**が明記されていれば、利用者はライセンスと法律の範囲内で自由に製作物を利用できる（もちろん、良識のない利用は慎むべきである）。

たとえば、本書は CC BY-NC-ND 4.0 というライセンスの下で配布されている。したがって、読者は自由に本書を再配布できる。しかし、本書を改変した場合は再配布できないなど、いくつかの制約が課される。

1.5 補遺

1.5.1 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 1.6 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、適切な文字コードを指定してテキストを開きなおせばよい。Visual Studio Code⁶⁾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 1.6: 文字化けしたテキスト

また、テキストを UTF-8 で符号化したにもかかわらず、テキストエディタが文字コードを ASCII と判定する場合がある。これは、UTF-8 と ASCII の仕様には共通する部分があるので、たまたまテキスト中の文字がすべて ASCII の文字集合にも属していた場合、文字コードをどちらとも解釈できることがあるからである。

1.5.2 ラスタ形式と拡大・縮小

ラスタ形式の画像に対する拡大・縮小は、元の画像から新たな画像を作成する操作と言える。そのため、拡大・縮小の手法は複数存在する。ニアレストネイバー法、バイリニア法、バイキュービック法は、よく知られた拡大・縮小の手法である。各手法で画像を拡大したときの様子を図 1.7 に示す。ただし、図 1.7 の画像編集はすべて GIMP⁷⁾で行った。また、各手法の特徴が分かりやすくなるように、(b) から (d) は (a) の画像を幅が 100px になるまでニアレストネイバー法で縮小してから、幅が 1600px になるまで拡大した。

図 1.7 において生クリームの境界線に注目すると、拡大結果がバイキュービック法、バイリニア

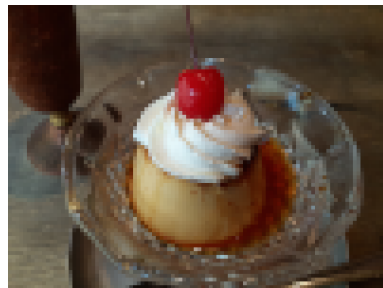
5) 適正な範囲での引用など、例外はある。[1] にこうした例外が列記されている。

6) <https://code.visualstudio.com>

7) <https://www.gimp.org>



(a) 元画像



(b) ニアレストネイバー法



(c) バイリニア法



(d) バイキュービック法

図 1.7: 画像の拡大手法の比較

法，ニアレストネイバー法の順になめらかであることが分かる。

図 1.7 で挙げた以外にも，画像が粗くなるのを回避するために機械学習を応用した手法が提案されていたりと [4]，様々な手法が研究されている。

第2章 $\text{T}_{\text{E}}\text{X}/\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ とは

この章では、 $\text{T}_{\text{E}}\text{X}/\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ の概略を説明する。

2.1 $\text{T}_{\text{E}}\text{X}$ と $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$

文字・図版などを配置し、紙面を設計する作業を**組版**という。 $\text{T}_{\text{E}}\text{X}$ は組版処理ソフトウェアの1つである。オリジナルの $\text{T}_{\text{E}}\text{X}$ は Donald Knuth により開発された。今日、オリジナルの $\text{T}_{\text{E}}\text{X}$ に新機能が追加されることは無いが、様々な拡張版が有志の手で開発されている。本書では、Knuth による最初の $\text{T}_{\text{E}}\text{X}$ 、およびその拡張を **$\text{T}_{\text{E}}\text{X}$ エンジン**と総称する。

$\text{T}_{\text{E}}\text{X}$ を利用する場合、通常は**マクロパッケージ**というものが併用される¹⁾。マクロパッケージは、 $\text{T}_{\text{E}}\text{X}$ エンジンが提供する機能を使いやすくするプログラムである。 $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ はマクロパッケージの1つであり、²⁾ 現在も開発が続いている。 $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X} 2_{\epsilon}$ は今日広く利用されているバージョンの $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ であり、本書も主に $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X} 2_{\epsilon}$ について扱う。

$\text{T}_{\text{E}}\text{X}$ に関する情報を知りたい場合は、**CTAN** (<https://ctan.org/>) を参照するとよい。CTAN は「The Comprehensive $\text{T}_{\text{E}}\text{X}$ Archive Network」の略であり、 $\text{T}_{\text{E}}\text{X}$ に関係するプログラム・文書・フォントなどのリソースを集約した Web サイトである。

2.2 $\text{T}_{\text{E}}\text{X}$ を利用する方法

$\text{T}_{\text{E}}\text{X}$ を利用するには、主に2つの方法がある。

1. $\text{T}_{\text{E}}\text{X}$ ディストリビューションを利用する
2. Web サービスを利用する

まず「1. $\text{T}_{\text{E}}\text{X}$ ディストリビューションを利用する」について説明する。 **$\text{T}_{\text{E}}\text{X}$ ディストリビューション**とは、 $\text{T}_{\text{E}}\text{X}$ に関連するリソースをまとめたソフトウェアである。

現在、日本では **$\text{T}_{\text{E}}\text{X}$ Live** という $\text{T}_{\text{E}}\text{X}$ ディストリビューションがよく使われている。 $\text{T}_{\text{E}}\text{X}$ Live は $\text{T}_{\text{E}}\text{X}$ エンジンだけでなく、フォントや、含まれるソフトウェアのマニュアルなども内包している（マニュアルの参照方法は小節 2.6.1 で紹介する）。インストール方法は、公式サイト (<https://www.tug.org/texlive/>)、および [3] にある。

次に「2. Web サービスを利用する」について説明する。**Overleaf** は、Web サイト上で $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ を利用できるサービスである。インストールの手間を省けるだけでなく、様々な機能を GUI で利用することが利点である。利用方法については、公式サイト (<https://www.overleaf.com/>) を参照し

1) 節 2.5 で説明する「パッケージ」とは別物である。

2) 他にも Plain $\text{T}_{\text{E}}\text{X}$, Con $\text{T}_{\text{E}}\text{X}$ t など、マクロパッケージは複数存在する。

てほしい。

なお、以下では $\mathrm{T}_{\mathrm{E}}\mathrm{X}$ Live がインストールされていると仮定する。

2.3 $\mathrm{T}_{\mathrm{E}}\mathrm{X}$ と和文

オリジナルの $\mathrm{T}_{\mathrm{E}}\mathrm{X}$ は和文の組版に対応していない。そのため、和文の組版に対応した $\mathrm{T}_{\mathrm{E}}\mathrm{X}$ エンジンが開発されてきた。 $\mathrm{pT}_{\mathrm{E}}\mathrm{X}$ と $\mathrm{upT}_{\mathrm{E}}\mathrm{X}$ はそうした、和文に対応した $\mathrm{T}_{\mathrm{E}}\mathrm{X}$ エンジンである。

$\mathrm{pT}_{\mathrm{E}}\mathrm{X}$ と $\mathrm{upT}_{\mathrm{E}}\mathrm{X}$ では、 $\mathrm{upT}_{\mathrm{E}}\mathrm{X}$ のほうが多種多様な文字を入力できる。正確には、 $\mathrm{pT}_{\mathrm{E}}\mathrm{X}$ は **JIS X 0208** という文字集合に対応しているのに対し、 $\mathrm{upT}_{\mathrm{E}}\mathrm{X}$ は Unicode の文字集合に属する日中韓の文字を扱える。たとえば、「①」を $\mathrm{upT}_{\mathrm{E}}\mathrm{X}$ では直接入力できるが、 $\mathrm{pT}_{\mathrm{E}}\mathrm{X}$ では工夫を要する³⁾。

この他にも、 $\mathrm{LuaT}_{\mathrm{E}}\mathrm{X}$ という $\mathrm{T}_{\mathrm{E}}\mathrm{X}$ エンジンでも和文を扱える。 $\mathrm{LuaT}_{\mathrm{E}}\mathrm{X}$ は、 $\mathrm{LuaT}_{\mathrm{E}}\mathrm{X}$ 自身を汎用プログラミング言語 **Lua** によって拡張できるようにした $\mathrm{T}_{\mathrm{E}}\mathrm{X}$ エンジンである。 $\mathrm{LuaT}_{\mathrm{E}}\mathrm{X}$ そのものは、そのままでは和文の組版に対応していない。しかし、 $\mathrm{LuaT}_{\mathrm{E}}\mathrm{X}\text{-ja}$ というパッケージ⁴⁾によって、和文の組版に対応させられる。

2.4 ドキュメントクラス

以下では $\mathrm{L}^{\mathrm{A}}\mathrm{T}_{\mathrm{E}}\mathrm{X} 2_{\epsilon}$ について解説する。 $\mathrm{L}^{\mathrm{A}}\mathrm{T}_{\mathrm{E}}\mathrm{X} 2_{\epsilon}$ の特徴は、文書の執筆者が組版についてあまり気にせずとも、適切なレイアウトがなされた文書を得られることである。これは、 $\mathrm{L}^{\mathrm{A}}\mathrm{T}_{\mathrm{E}}\mathrm{X} 2_{\epsilon}$ の**ドキュメントクラス**と**パッケージ**という機構によって実現されている。

まず、ドキュメントクラスについて説明する。`test.tex` というファイルを以下の内容で作成⁵⁾し、適当なディレクトリに置く。

```
\documentclass[uplatex,dvipdfmx,30pt]{jsarticle}
\begin{document}
  試しに数式を入力してみる。
  \begin{equation}
    (x^n)' = nx^{n-1}
  \end{equation}
\end{document}
```

次に、そこをカレントディレクトリとして、次のコマンドを実行する。このコマンドにより、マクロパッケージが読み込まれた状態で $\mathrm{T}_{\mathrm{E}}\mathrm{X}$ エンジンが起動する⁶⁾。

```
> uplatex test
```

3) この工夫については [7] を参照のこと。

4) パッケージについては節 2.5 で改めて述べる。

5) テキストエディタはメモ帳でよい。

6) このとき起動する $\mathrm{T}_{\mathrm{E}}\mathrm{X}$ エンジンは、 $\epsilon\text{-upT}_{\mathrm{E}}\mathrm{X}$ という $\mathrm{upT}_{\mathrm{E}}\mathrm{X}$ とは少し異なる $\mathrm{T}_{\mathrm{E}}\mathrm{X}$ エンジンである。このことは小節 2.6.3 にもう少し詳しく書いた。

すると、同じディレクトリに `test.dvi` というファイルが出力される。**DVI** は、 $\text{T}_{\text{E}}\text{X}$ エンジンによって組版の指示が書き込まれたファイルのファイル形式である⁷⁾。指示に沿った PDF を作成するには、`DVIPDFMx` というプログラムを利用する。すなわち、次のコマンドを実行する。

```
> dvipdfmx test
```

これにより、DVI が PDF へと変換され、`test.pdf` というファイルが同じディレクトリに出力される。出力例を図 2.1 に示す。

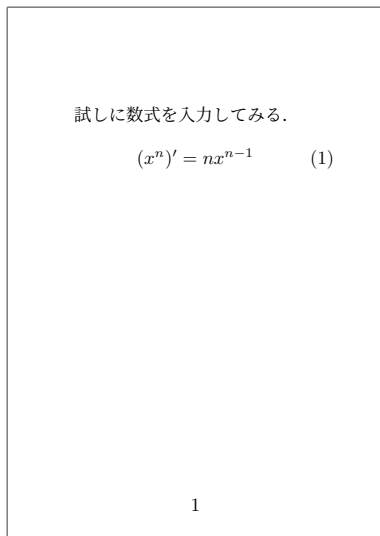


図 2.1: up $\text{L}_{\text{A}}\text{T}_{\text{E}}\text{X}$ の出力例

図 2.1 を見ると、 $\text{L}_{\text{A}}\text{T}_{\text{E}}\text{X}$ ソースにおいては余白・フォントなどを指定していないにもかかわらず、出力された PDF では適切な組版が行われていることが分かる。

これは、ソースの 1 行目「`\documentclass{jsarticle}`」が、そうした設定を代わりに行っているためである。より正確には、この行は「`jsarticle` というドキュメントクラスを利用する」ことを宣言している。この宣言を読み取った $\text{T}_{\text{E}}\text{X}$ エン진은、`jsarticle.cls` というファイルを読み込む。`jsarticle.cls` には余白・フォントなどがあらかじめ指定されているので、ユーザが 1 つ 1 つ設定せずとも、適切に組版が行われる。

2.5 パッケージ

前節に続いて、パッケージについて説明する。たとえば、記号「 \angle 」を使いたいとする。この記号は、 $\text{L}_{\text{A}}\text{T}_{\text{E}}\text{X} 2_{\epsilon}$ で初めから用意されているわけではない。また、`jsarticle` もこの記号を提供していない。

7) すべての $\text{T}_{\text{E}}\text{X}$ エンジンが DVI を出力するわけではない。実際、 $\text{p}_{\text{L}}\text{T}_{\text{E}}\text{X}$ 、 $\text{up}_{\text{L}}\text{T}_{\text{E}}\text{X}$ は DVI を出力するが、 $\text{Lua}_{\text{L}}\text{T}_{\text{E}}\text{X}$ は PDF を直接出力する。

このようなときは、パッケージを利用して記号を追加するとよい。具体的には、「`\documentclass [...]{...}`」と「`\begin{document}`」の間に「`\usepackage{amssymb}`」を追記する。

```
\documentclass[uplatex,dvipdfmx,30pt]{jsarticle}
\usepackage{amssymb}
\begin{document}
  試しに数式を入力してみる.
  \begin{equation}
    \measuredangle
  \end{equation}
\end{document}
```

2 行目「`\usepackage{amssymb}`」を読み取った \TeX エン진은、`amssymb.sty` というファイルを読み込む。`amssymb.sty` では、`\measuredangle` というコマンドが「 \sphericalangle 」を出力するように定義されている。そのため、この \LaTeX ソースを前節と同様に PDF へと変換すると、図 2.2 に示す出力が得られる。

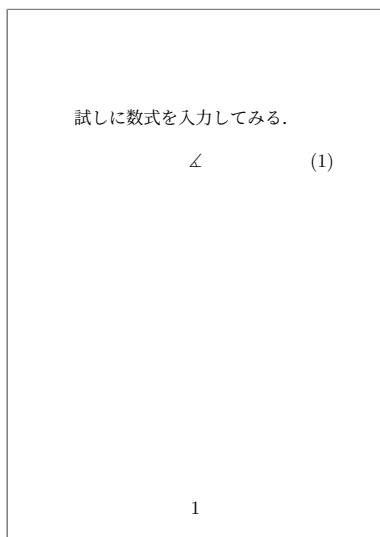


図 2.2: パッケージによる記号の追加

`amssymb` のほかにも、余白を変更したり、グラフを作成したりと、様々なパッケージが \TeX Live には含まれている。たとえば [2] に、よく使われるパッケージが記載されている。

2.6 補遺

2.6.1 Texdoc

$\mathrm{T}_{\mathrm{E}}\mathrm{X}$ Live をインストールすると、 $\mathrm{T}_{\mathrm{E}}\mathrm{X}$ Live に含まれるリソースと関係する文書も同時にインストールされる。それらの文書は **Texdoc** というプログラムを介して参照できる。たとえば、コマンドラインで次のプログラムを実行すると、 $\mathrm{T}_{\mathrm{E}}\mathrm{X}$ Live のガイドを参照できる。

```
> texdoc texlive
```

コマンドラインに不慣れでも、Texdoc のオンライン版 (<http://texdoc.org/>) を利用できる。本書を読むにあたって、適宜 Texdoc を参照してほしい。

2.6.2 文書作成を支援するソフトウェア

$\mathrm{T}_{\mathrm{E}}\mathrm{X}$ で文書を作成するには、テキストファイルを作成して $\mathrm{T}_{\mathrm{E}}\mathrm{X}$ エンジンを実行すればよい。したがって、Windows に付属するメモ帳とコマンドプロンプトだけでも、 $\mathrm{T}_{\mathrm{E}}\mathrm{X}$ で文書を作成することはできる。しかし、通常はより快適に文書を作成するため、文書作成を支援するソフトウェアを利用する。

$\mathrm{T}_{\mathrm{E}}\mathrm{X}$ Live をインストールしていれば、TeXworks⁸⁾ というソフトウェアがインストールされている。TeXworks を利用すると、 $\mathrm{T}_{\mathrm{E}}\mathrm{X}$ ファイルの編集と $\mathrm{T}_{\mathrm{E}}\mathrm{X}$ エンジンの起動を GUI で行える。

また、本書は LaTeX Workshop⁹⁾ と llmk¹⁰⁾ を利用して作成されている。前者は $\mathrm{L}^{\mathrm{A}}\mathrm{T}_{\mathrm{E}}\mathrm{X}$ ソースの編集を、後者は $\mathrm{T}_{\mathrm{E}}\mathrm{X}$ エンジンに関するプログラムの実行を、それぞれ支援するソフトウェアである。

2.6.3 $\varepsilon\text{-}\mathrm{T}_{\mathrm{E}}\mathrm{X}$

実は、 $\mathrm{upL}^{\mathrm{A}}\mathrm{T}_{\mathrm{E}}\mathrm{X}$ を起動したときに実行されるのは、 $\varepsilon\text{-}\mathrm{upT}_{\mathrm{E}}\mathrm{X}$ という $\mathrm{upT}_{\mathrm{E}}\mathrm{X}$ とは異なる $\mathrm{T}_{\mathrm{E}}\mathrm{X}$ エンジンである。実際、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.
**
```

$\varepsilon\text{-}\mathrm{upT}_{\mathrm{E}}\mathrm{X}$ は、 $\varepsilon\text{-}\mathrm{T}_{\mathrm{E}}\mathrm{X}$ という $\mathrm{T}_{\mathrm{E}}\mathrm{X}$ エンジンの拡張であることが $\mathrm{upT}_{\mathrm{E}}\mathrm{X}$ と異なる¹¹⁾。本書では、 $\varepsilon\text{-}\mathrm{T}_{\mathrm{E}}\mathrm{X}$ 拡張の有無についてはいちいち断らないことにする。

2.6.4 他の命令の引数にできない命令

$\mathrm{T}_{\mathrm{E}}\mathrm{X}$ の仕様により、`\verb` など一部の命令は、他の命令の引数にできない。たとえば、次の $\mathrm{L}^{\mathrm{A}}\mathrm{T}_{\mathrm{E}}\mathrm{X}$ ソースは `\verb` が `\footnote` の引数になっているので不正である。

8) <http://www.tug.org/texworks/>

9) <https://github.com/James-Yu/LaTeX-Workshop>

10) <https://github.com/wtsnjp/llmk>

11) 現在利用される $\mathrm{T}_{\mathrm{E}}\mathrm{X}$ エンジン ($\varepsilon\text{-pT}_{\mathrm{E}}\mathrm{X}$ ・ $\varepsilon\text{-upT}_{\mathrm{E}}\mathrm{X}$ ・ $\mathrm{pdfT}_{\mathrm{E}}\mathrm{X}$ など) の多くは $\varepsilon\text{-}\mathrm{T}_{\mathrm{E}}\mathrm{X}$ の拡張である。

```
\documentclass[uplatex,dvipdfmx]{jsarticle}
\begin{document}
\TeX のロゴ (\emph{\verb!\TeX!で出力できる}) は, Donald Knuthがこう表記するよう
に求めている.
\end{document}
```

実際, `example.tex` を上記の内容で作成し, 次のコマンドを実行すると「LaTeX Error: `\verb` illegal in command argument.」というエラーが出力される.

```
> uplatex -kanji=utf8 -no-guess-input-enc example
```

この問題を解決する手っ取り早い方法は, `\verb` を使わないことである. すなわち, `\` を `\textbackslash` に置き換えればよい.

```
\documentclass[uplatex,dvipdfmx]{jsarticle}
\begin{document}
\TeX のロゴ (\emph{\texttt{\textbackslash TeX}で出力できる}) は, Donald Knuthが
こう表記するように求めている.
\end{document}
```

第 3 章 Lua \LaTeX による実例

この章では、Lua \LaTeX による文書作成の実例を示す。ただし、Lua \LaTeX とは \TeX エンジン Lua \TeX とマクロパッケージ \LaTeX の組み合わせを指す。

なお、文章を組版する方法については [6] に、数式を組版する方法については [9] と [10] に、それぞれ詳しく記されている。また、検索すれば多くの Web サイトがヒットする。そこで本書では、最低限の説明に加えて、和文特有の注意点を述べるにとどめる。

また、[2] は参考になる文書を目的別に整理している。ごく簡潔であるので、この章を読む前に目を通すことを推奨する。本書のソースは GitHub で公開されているので、それも参考にできるだろう。

3.1 和文のドキュメントクラス

Lua \LaTeX で利用でき、和文の文書作成を目的とするドキュメントクラスには、大きく次の 3 系統がある。

1. Lua \LaTeX -ja 用 jsclasses 互換クラス (<https://osdn.net/projects/luatex-jaj/>)
2. BXjscls (<https://github.com/zr-tex8r/BXjscls>)
3. jlreq (<https://github.com/abenori/jlreq>)

なお、ドキュメントクラスが Lua \LaTeX に対応していれば、`luatexja` パッケージを読み込むことで和文の組版に対応させられる。

本書では「1.Lua \LaTeX -ja 用 jsclasses 互換クラス」を利用した例を示す¹⁾。Lua \LaTeX -ja 用 jsclasses 互換クラスとは、`p \LaTeX` と `up \LaTeX` で利用できた jsclasses というドキュメントクラス群を、Lua \LaTeX でも利用できるようにしたドキュメントクラス群である。`ltjsarticle`、`ltjsreport` などがこれに属する。

実際に使ってみる。次の内容で `test.tex` というファイルを作成する²⁾。

```
\documentclass[17pt]{ltjsarticle}
% (from)
\title{表題}
\author{作者名}
\date{\today}
% (to)
```

1) なお、本書そのものは `jlreq` を利用して作成されている。

2) 1 行目の「17pt」という指定は、見本を本書に含めるときに、文字が小さくなりすぎるのを防ぐためのものであり、消して構わない。消す場合、1 行目は「`\documentclass{ltjsarticle}`」となる。

```

\begin{document}
\maketitle
\tableofcontents
\section{節見出し}
\subsection{小節見出し}
ここに本文が入る\footnote{脚注}.

```

段落を変えたければ、1つ空白行を設ける。

```

\end{document}

```

なお、 $\text{T}_{\text{E}}\text{X}$ では % から始まる行はコメントとして扱われ、出力に影響しない。また、% (from) から % (to) まで、すなわち `\documentclass[...]{...}` から `\begin{document}` の手前までを **プリアンブル** という。 `\usepackage` などはプリアンブルに記述する。

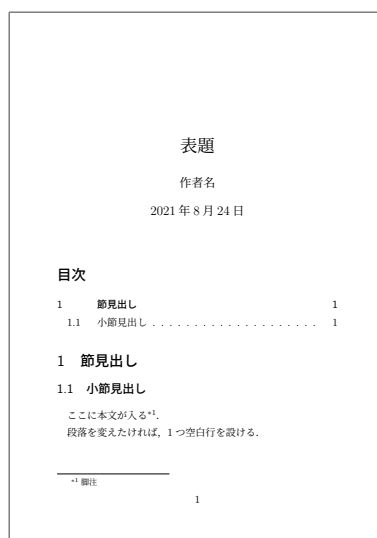
次に、コマンドラインから $\text{LuaL}_{\text{A}}\text{T}_{\text{E}}\text{X}$ を実行する。

```
> lualatex test
```

すると、図 3.1a に示す PDF が得られる。図 3.1a を見ると、目次の内容が書き込まれていないことが分かる。その代わり、PDF があるディレクトリに `test.toc` というファイルが生成されている。このファイルには、 $\text{L}_{\text{A}}\text{T}_{\text{E}}\text{X}$ ソースから構成された目次が書き込まれている。そこで、もう一度 $\text{LuaL}_{\text{A}}\text{T}_{\text{E}}\text{X}$ を実行すれば、目次が書き込まれた PDF を得られる。図 3.1b に完全な PDF を示す。



(a) $\text{LuaL}_{\text{A}}\text{T}_{\text{E}}\text{X}$ を 1 回実行した結果



(b) $\text{LuaL}_{\text{A}}\text{T}_{\text{E}}\text{X}$ を 2 回実行した結果

図 3.1: シンプルな和文の文書

3.2 文献管理と索引作成

通常、 $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ で参考文献を管理するには $\text{BibT}_{\text{E}}\text{X}$ というプログラムが用いられる。また、索引を作成するには `makeindex` というプログラムが用いられる。しかし、これらはいずれも和文と Unicode に対応していない。そこで、和文の文書を作る際には、代わりにこれらの拡張版を利用するとよい。 $\text{BibT}_{\text{E}}\text{X}$ については `upBibTEX`, `makeindex` については `upmendex` が、和文と Unicode に対応した拡張である。

3.3 $\text{LuaT}_{\text{E}}\text{X}$ によるフォント設定

$\text{LuaT}_{\text{E}}\text{X}$ において、欧文フォントの変更は `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プ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³⁾に変更している。

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

また、和文フォントの変更は `luatexja-fontspec`, あるいは `luatexja-preset` パッケージによって行う。

なお、フォントを変更するときは、フォントのライセンスを確認すべきである。上記のパッケージを利用すると、初期設定では PDF にフォントファイルの一部が埋め込まれる。これは、設定したフォントが読者のコンピュータにインストールされていなくても、PDF を適切に表示するためである。しかし、埋め込みの可否はフォントによって異なる。白源は **SIL Open Font License** というライセンスの下で配布されており、このライセンスは利用者に埋め込みを許可している。

3.4 グラフの作成

本書では、次の 2 つの方法を紹介する。

1. `PGFPlots` による方法
2. `GNUPlot` による方法

PGFPlots は、`TikZ` を利用してデータを可視化するパッケージである。利用方法はマニュアル [5] に詳しく記されているので、ここでは図 3.2 に実例を示すにとどめる。

```
\begin{figure}[htbp]
\centering
\begin{tikzpicture}
\begin{axis}[width=7cm]
```

3) <https://github.com/yuru7/HackGen>

```

\addplot [smooth,samples=100,domain=-pi:pi] {cos(deg(x))};
\addlegendentry{$\cos(x)$};
\end{axis}
\end{tikzpicture}
\caption{PGFPlotsによる $\cos(x)$ のグラフ}
\end{figure}

```

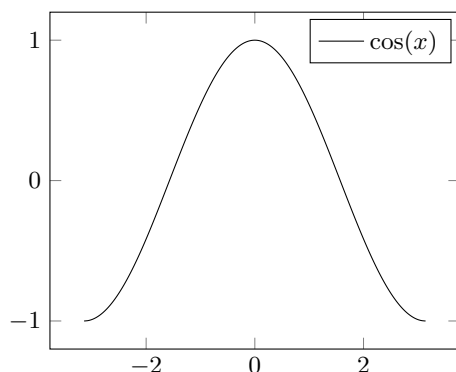


図 3.2: PGFPlots による $\cos(x)$ のグラフ

GNUPlot は、関数やデータを描画するプログラムである。公式サイト (<http://www.gnuplot.info/>) からダウンロードできる。こちらも、利用方法は公式サイトに詳しく記されているので、図 3.3 に実例を示すにとどめる。

まず、 \LaTeX 文書があるディレクトリに移動する。その後、コマンドラインで GNUPlot を起動し、以下のように入力する⁴⁾。

```

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

```

すると、カレントディレクトリにいくつかのファイルが出力される。プリアンプルに `\usepackage{gnuplot-lua-tikz}` を追加し、 \LaTeX 文書に次の記述を加える。すると、図 3.3 のグラフが得られる。

```

\begin{figure}[htbp]
\centering
\includegraphics{gnuplot.pdf}
\caption{GNUPlotによる $\cos(x)$ のグラフ}
\end{figure}

```

4) 1行目を `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。

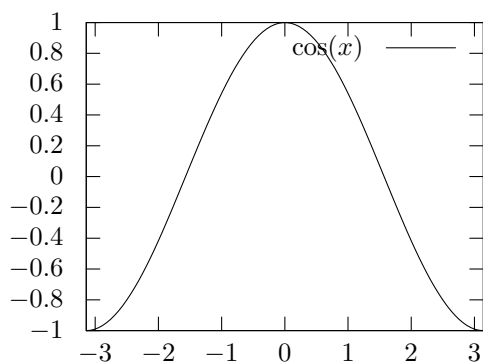


図 3.3: Gnuplot による $\cos(x)$ のグラフ

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}
```

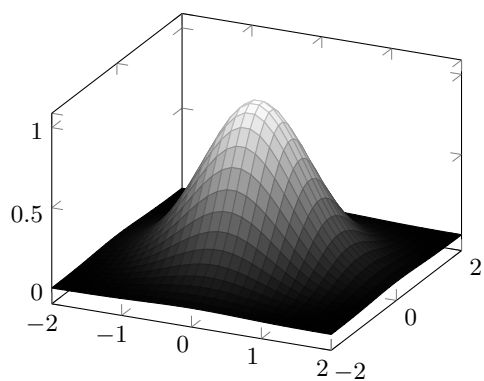


図 3.4: $e^{-(x^2+y^2)}$ のグラフ

参考文献

- [1] 著作物が自由に使える場合 | 文化庁. https://www.bunka.go.jp/seisaku/chosakuken/seidokaisetsu/gaiyo/chosakubutsu_jiyu.html, Accessed: 2021-08-23.
- [2] Karl Berry, Manuel Pégourié-Gonnard, and Jim Hefferon. A first set of L^AT_EX resources. <https://ctan.org/pkg/latex-doc-ptr?lang=en>, 2021. Accessed: 2021-07-17.
- [3] Karl Berry, 朝倉卓人. T_EX live ガイド 2021. <https://github.com/wtsnjp/texlive-ja>, 2021. Accessed: 2021-08-16.
- [4] Chao Dong, Chen Change Loy, Kaiming He, and Xiaoou Tang. Learning a deep convolutional network for image super-resolution. In David Fleet, Tomas Pajdla, Bernt Schiele, and Tinne Tuytelaars, editors, *Computer Vision – ECCV 2014*, pp. 184–199, Cham, 2014. Springer International Publishing.
- [5] Dr. Christian Feuersänger. Manual for package pgfplots: 2d/3d plots in L^AT_EX, version 1.18.1. <https://sourceforge.net/projects/pgfplots/>, 2021. Accessed: 2021-08-14.
- [6] Tobias Oetiker, Hubert Partl, Irene Hyna, and Elisabeth Schlegl. The not so short introduction to L^AT_EX 2_ε. <https://ctan.org/tex-archive/info/lshort/english/?lang=en>, 2021. Accessed: 2021-07-17.
- [7] Takuji TANAKA. uptex, uplatex — 内部 unicode 版 ptex, platex の実装. https://github.com/texjporg/uptex-base/blob/master/01uptex_doc_utf8.txt, 2020. Accessed: 2021-08-18.
- [8] 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.
- [9] アメリカ数学会. はやわかり L^AT_EX で数式組版. https://www.latex-project.org/help/documentation/short-math-guide_jpn.pdf, Accessed: 2021-08-24.
- [10] アメリカ数学会, L^AT_EX3 Project. amsmath パッケージユーザガイド (version 2.1) . https://www.latex-project.org/help/documentation/amslatex_jpn.pdf. Accessed: 2021-08-24.

索引

B

BibTeX 17

C

ConTeXt 9

CTAN 9

D

DVI 11

DVIPDFMx 11

F

fontspec 17

G

GNUPlot 18

J

JIS X 0208 10

JPEG 5

jsclasses 15

L

L^ATeX 9

L^ATeX 2_ε 9

LaTeX Workshop 13

llmk 13

ltjsarticle 15

ltjsreport 15

Lua 10

LuaTeX 10

LuaTeX-ja 10

luatexja-fontspec 17

luatexja-preset 17

M

makeindex 17

O

Overleaf 9

P

PGFPlots 17

Plain T_EX 9

PNG 5

pT_EX 10

S

SIL Open Font License 17

SVG 5

T

T_EX 9

Texdoc 13

T_EX Live 9

TeXworks 13

T_EX エンジン 9

T_EX ディストリビューション 9

U

Unicode 4

upBibTeX 17

upmendex 17

upT_EX 10

V

Visual Studio Code 7

か

可逆圧縮 6

画素 5

く

組版 9

と

ドキュメントクラス 10

に

ニアレストネイバー法 7

は

バイキュービック法	7
バイリニア法	7
パッケージ	10

ひ

非可逆圧縮	6
-------	---

ふ

符号位置	4
符号化文字集合	4
符号空間	4
符号単位	4
プリアンブル	16

へ

ベクタ形式	5
-------	---

ま

マクロパッケージ	9
----------	---

も

文字コード	4
文字集合	4
文字符号化形式	4
文字符号化スキーム	5

ら

ライセンス	7
ラスタ形式	5