README_JP.pdf

連絡先: bin4tsv@gmail.com

2018 年 6 月 7 日 (木) – 第 1 版

この文書は、CPAN にアップロードされたモジュール CLI::LaTeX::Table について、インストール方法などを解説する。他のマニュアルを本モジュールのインストールの方法など (出来れば実行のさせ方を含め)、作成者の意図がきちんと伝わるように、この文書で説明を試みる。ただし本モジュールの作成者のモジュール配布の経験は半年以下であり、間に合わせの知識でこの文書を作成しており、さらに良い方法は大いにあり得る。

目次

1 想定されている計算機環境

Perl 5.14 以降が使える環境であれば、基本的にはどんな環境でも使えるように、本モジュール CLI::LaTeX::Table は作られている。Unix でも Linux でも Windows でも、Mac OS X でも使えるが、本

文書作成者は、Mac OS X 上の Unix 環境及びクラウドサービスの AWS (Amazon Web Service) の EC2 の Linux 端末で検証を行っている。bash と zsh のシェルで検証をしているが、それ以外の csh などのシェルでは読者が下記の文書の意図を読み取って独自調査をしてインストール及び利用をする必要がある。(本書は、「全く知識が無くてもすぐ使えるマニュアル」にはなっていない。「初心者であってもキーワードを元にインターネットで調べたり試行錯誤を何度か繰り返せば為になる意味の分かる手順書」「いくらか知識のある人がさっと読んでも要領の理解しやすい手順書」になっていることを意図して書かれている。)

2 インストールの方法

下記のインストールの方法のどれかが完了したら、次の節の「使い始める」を参照のこと。

2.1 インストール前の注意点

インストール方法の選択:

本モジュールのインストールの方法は、以下のどれか 1 つであり、利用者の状況に応じて選択する。[*1]

1. cpan を使う方法
2. cpanm を使う方法
3. Module::Starter の Build を使う方法
4. scripts のディレクトリ下のプログラムにパスを通す方法

上記 4 通りのインストールの方法を決めるには、表 1 による比較が参考になるであろう。**cpan** がインストールされていない場合それをインストールするには通常 root 権限 (管理者権限) を必要とするし、そのインストールには数分間画面を注視するような 作業を要する。近年、**cpanm** の利用も多いと思われるが、それが未インストールの場合は cpanm のインストール作業が数分間必要である。またこれら適切に使いこなすには[*2]、様々な知識の習得があらかじめ必要と考えられる。Module::Starter の **Build** を使う方法と**パス**を通す

本モジュールをインストールする方法 4 個の比較

本 module の install 法→	cpan コマンドによる方法	cpanm コマンドの方法	Build の方法	手作業展開とパス設定
作業前に必要な環境:	cpan install 済みの環境	cpanm intall 済みの環境	Perl のみ **:-)**	Perl のみ **:-)**
その環境の準備の手間:	cpan の初期設定に数分間	cpanm の install	皆無	皆無 **:-)**
管理者 (root) 権限の必要性:	必要がある場合が多い	install 先 directory による	install 先 directory による	不要 **:-)**
install の手間:	cpan module 名 で一発 **:-)**	cpanm module 名 で一発 **:-)**	**download, 解凍, コマンド数回**	左に加え、**PATH 設定作業**
ありがちな追加の手間:	設定が大変な場合がある	設定が大変な場合がある	関連 module の install が発生	PATH 設定コマンドをどこに書くか
install する directory の指定方法:	cpan 起動して o conf など	cpanm -l abcde	./Build install --intall_base abcde	手作業; mkdir など
アンインストールの方法:	やや煩雑	cpanm -U (実験的)		PATH 設定の書き加えの除去など
計算機環境の破損の危険:	root 権限 install であり得る	root 権限 install であり得る	root 権限 install であり得る	手作業が多いので起こりにくい
既存のコマンド上書きの危険:	あり得る	あり得る	あり得る	手作業が多いので起こりにくい
既存コマンドと名前衝突:	あり得る	あり得る	あり得る	手で名前を書き換えて回避可能

表 1　この表の情報は確実とは限らない。最後の 3 行は誇張の可能性もある。

[*1] どれを採用するのが最適であるかについては、利用者の計算機環境での権限、本モジュールを迅速にインストールしたいかどうか、計算機環境の変更を最小限にしたいかどうか、本モジュールを試しに使うかもしくは恒久的に使うか、利用者の外部ライブラリの管理ポリシーなどに依存する。

[*2] たとえば次の方法を **cpan** 及び **cpanm** について知ることが望ましい: 計算機環境に不具合を起こさないかテストする方法、ユー

方法は、利用者が metacpan のサイトから本モジュールをダウンロードして、tar xzvf または tar -xzvf などで解凍する作業を伴う。

インストール直後に記録/記憶すべきこと:
どの方式でインストールをしたか、どの権限でインストールしたか、どこのディレクトリにプログラムをインストールするかを設定した場合のその場所は、必要な場合に思い出せるようにする必要はある。追加インストールもしくアンインストールに必要があるためである。(ただしそれを忘れても、簡便な調査で後で調べることは可能。)

2.2 インストールの手順

この節で下記は 主に Unix/Linux 環境を仮定する。しかし他の OS であっても十分に参考になるであろう。

2.2.1 scripts のディレクトリ下のプログラムに直接手でパスを通す場合

”手作業”で本モジュールをインストールする方法:

1. metacpan などのサイトから **ダウンロード**。(Download と書かれた項目をブラウザ上で探してクリックする。)
2. **解凍する**。ファイル名 *.tar.gz を対象に tar xzvf または tar -xzvf を実行することで解凍は可能。
3. 解凍先のディレクトリに入って (cd コマンド)、scripts 直下のファイルを確認 (ls コマンド) :
   - これからコマンドとして使うファイル名なので、後の必要に応じてすぐ把握可能とするため。
   - (必要に応じ) init.sh 以外の各ファイルのパーミッションが実行可能に設定されていることを確認。[*3]
   - 他の実行可能ファイルと名前が**衝突**することがないかどうかを後で必要に応じ確認可能とするため。[*4]
4. scripts 直下の全ファイルの新たな**格納先のディレクトリ**を決める (例: ～/bin や /home/you/bin4tsv)。
   - そのコピー先のディレクトリがまだ無い場合は、 mkdir コマンドで作成。
   - そのコピー先に既に他のファイルが存在する場合は、scripts 下のファイルと名前が衝突しないかを確認する。衝突する場合は、scripts 直下のファイル名を適宜変更する (mv -i コマンド)。
5. **パス PATH の設定:** (格納されたディレクトリにあるファイルが今後必要なときにすぐ実行出来るようにする。)
   - ここで前記で決めた格納先のディレクトリを以下、 somewhere とする。
   - Windows の場合は、**システム環境変数** の Path の値を編集し、somewhere を書き足す。[*5]
   - Unix/Linux の OS の場合: 使っているシェルに応じて、Bash ならば ～/.bash_profile に、Zsh ならば ～/.zshrc に、次の行を (ログイン時に毎回実行されるように) 書き加える : [*6]

     `source somewhere/init.sh`
   - 一時的に PATH を設定する場合: Bash または Zsh の場合、次を実行。[*7]: `PATH=$PATH:`somewhere

[*3] ザー権限のみあれば実行可能なローカル環境にインストールする方法、アンインストールの方法。

2.2.2 Module::Starter の Build を使う場合

1. metacpan などのサイトから **ダウンロード**。(Download と書かれた項目をブラウザ上で探してクリック)。
2. **解凍する**。ファイル名 *.tar.gz を対象に tar xzvf または tar -xzvf を実行することで解凍は可能。
3. 解凍先のディレクトリに入って (cd コマンドを使う)、下記を順に実行。
   (1) perl Build.PL
   (2) ./Build
   (3) ./Build test
   (4) ./Build install

2.2.3 cpanm を使う場合

cpanm CLI::LaTeX::Table で完了する。cpanm の設定によってはローカル (/perl5/bin/など) にインストールされる。root のパスワードを要求されるが cpanm -S CLI::LaTeX::Table もしくは sudo cpanm CLI::LaTeX::Table を使う場合もある。

2.2.4 cpan を使う場合

cpan CLI::LaTeX::Table で完了する。もしくは sudo cpan CLI::LaTeX::Table で完了する。

2.3 アンインストールの方法

cpan, cpanm もしくは Build の方法を使った場合には、それぞれの方法のアンインストールの方法が利用できる。(本モジュールの作成者は現状、このやり方についてここでは触れない。) 手作業でプログラムにパスを通す方法でインストールをした場合は、上記の作業で行ったことを元に戻す。

2.4 依存する他のモジュールのインストール

本モジュール CLI::LaTeX::Table は csv2tsv というプログラムがあり、このプログラムはコアモジュールではない CPAN モジュール Text::CSV_XS に依存する。これがインストールされていない場合は、cpanm または cpan でインストールする。(Text::CSV_XS は既に広く使われているので、この方法で問題が起きることは考えにくい。)

---

*3 ls -l コマンドの出力において、空白区切りの 1 列目で r,w,x のなどの文字の配列を確認する。r は読み取り可能、w は書込可能、x は実行可能を意味する。rwxrwxrwx や r-xr-xr-x のような文字列において、これら 9 文字は 3 文字ごとに左から、ユーザーの権限、グループの権限、その他の人の権限を意味する。この 9 文字の前後に、ディレクトリであるか否か、リンクファイルであるか否かを表す文字が付加される場合がある。

*4 which -a プログラム名 で検証可能。後の作業で既存ファイルへの上書き回避と、パス上の実行優先順位の検討が可能。

*5 セミコロン (;) で somewhere を後ろに追加する。システム環境変数の設定ができるようにするには、Windows 7 の場合は、「システムのプロパティ」を起動して「詳細設定」のメニューを選び「環境変数」のボタンを押す。(なお、システム環境変数の設定の方法は、インターネットで検索すれば容易に発見できる。)

*6 使っているシェルを確かめる場合は echo $0 を実行する。

*7 = の前後に空白文字を含めてはいけない。

3 各コマンドを使い始める

3.1 saikoro 一様乱数

saikoro は一様乱数を出力する。コマンドラインで下記を順次実行する。

saikoro → 乱数が表示される。サイコロのように 1 から 6 までの整数が出力される。
saikoro > /dev/null → 標準エラー出力のみが表示される。
saikoro -g 5x3 → 5 行 3 列の乱数が表示される。
saikoro -g 10x5 -y 91..100 → 91 から 100 の範囲の整数の一様乱数が、10 行 5 列の乱数で出力。
saikoro -s 123 → 乱数シードの設定。再現性が確保される (繰り返し実行しても同じ結果になる)。
saikoro -. 3 → 小数点以下 3 桁まで表示する。連続一様乱数になる。
saikoro --help → ヘルプマニュアルの表示
saikoro --help opt → ヘルプの内、オプションスイッチのみを表示。
saikoro --help en → 日本語では無く、英語のヘルプを表示。

saikoro -/ , -g 5x5 → 横方向の区切り文字の変更。CSV 形式として使える。
saikoro -g12x12 -y 1.5e5 → 科学的表記法 (1.5e5) も使える。
saikoro -g12x12 -y 1.5e5 | less -x7 → タブ間隔は less で表示中も -x N enter で変更可能。
tabs -6 → タブの表示間隔を 6 文字に変更する。元に戻す場合は、単に tabs を実行。

3.2 transpose 転置行列

transpose は標準入力から TSV 形式 (タブ文字区切り形式) で行列を読取り、その転置行列 (縦方向と横方向を逆転した行列) を標準出力に出力する。入力は数値で無くても良い。

transpose <(saikoro -s123) → <(..) はプロセス置換である。
saikoro -s123 -g3x5 | transpose → saikoro -s123 -g3x5 と比較せよ。
saikoro -s56 -g 7x8 | transpose
saikoro -/, -g3x5 | transpose -/ ";" → トリッキーな使い方なので、汎用性はあまり無い。

3.3 csel 列選択

csel は簡単な列処理を AWK 言語や cut コマンドよりも簡単に実行出来るように考えて作られた。標準出力に出力する。入力は数値で無くても良い。

saikoro -q -y23 -g4x5 -s67 > tmp01 ← 一時的なファイル tmp01 にデータを書き込む。
csel -p4,2 tmp01 ← 4 列目と 2 列目のみを出力。上記のと比較せよ。
csel -p4,2 < tmp01 ←リダイレクション (<)
< tmp01 csel -p4,2 ← 似た様なコマンドの末尾のみを変更したい場合、便利。
csel -d 2,4 tmp01 ← 2 列目と 4 列目の出力を抑制。

`csel -d 2..4 tmp01` ← 2 列目から 4 列目までの出力を抑制。
`csel -d -1 tmp01` ← 最右列の出力を抑制
`csel -h -1 tmp01` ← 最右列を最も左 (先頭 head) に出力。
`csel -t -2,3 tmp01` ← 2,3 列目を最も右 (末尾 tail) に移動。
`csel -t -2,3 tmp01 | csel -~ -t 2,3` ← 列の並びを元に戻す。何か処理を挟む場合に便利。

3.4 latextable LaTeX 用製表機

latextable は、エクセルや各種 SQL ソフトの出力結果から LaTeX の表 (table 環境) への使い易く高機能な変換器として作られた。様々な記号を含む文字列を LaTeX に使えるようにすぐ変換する目的に使うこともできる。本モジュール CLI::LaTeX::Table の中心となる機能である。

```
saikoro -g12x12 | latextable ← LaTeX を知っているなら何が起きたか分かるはず。
latextable ← この 1 つの単語を端末で入力して (Enter を押下する)。
             そしてキーボードで何かを入力し最後に Ctrl+D を押下。
             すると \begin{table} と \end{table} で囲まれた文字列が生成される。
             要領がつかめたら、LaTeX の .tex 形式のファイルに記入して動作を確かめる。
             エクセルや各種表計算ソフトからコピーをしてペーストをして、いろいろ試すと良い。
latextable --help opt ← どのようなスイッチオプションが使えるか確かめる。
latextable -s ← 入力の余計な空白を取り去る。(BigQuery をブラウザ経由で使う場合に便利。)
latextable -_ ← 半角と全角の空白、全角ハイフンの存在を分かり安く表示するようにする。
latextable -3 コンマ区切り 1 始まりの列番号 ← 数値を 3 桁毎に区切って右寄せする。
latextable -j ← 日本語の半角カナを半角の幅で表示するようにする。
latextable -m0.7 -x40mm ← 大きな表を 0.7 倍の大きさにし、センタリングをする時に使う。
latextable -C1 ← 任意の (UTF8 の) 文字列を、LaTeX 用に表とは関係無く簡潔に出力。
latextable -Cmn ← table 環境の caption で複数行にまたがる SQL 文を挿入する時便利。
```

3.5 csv2tsv CSV 形式→ TSV 形式

csv2tsv は CSV 形式 (RFC4180) を TSV 形式に変換する。このプログラムは コアモジュールでは無い CPAN モジュール Text::CSV_XS に依存している。(似た様な実装は容易に他の人も思いつくので、プログラム名が衝突する可能性は、本モジュールの他の 4 個のプログラムに比べ、最も高いと考えられる。)

```
csv2tsv foo.csv > foo.tsv  ← CSV 形式の foo.csv を読取り TSV 形式に変換。
csv2tsv  もしくは cat | csv2tsv  ← 手で CSV 形式で入力する。
csv2tsv -t '(TAB)'  ← CSV 形式のあるレコードがタブ文字を含む場合、何に変換するか指定。
csv2tsv -n '(ENTER)'  ← C あるレコードが改行文字を含む場合、何に変換するか指定。
csv2tsv -2  ← 出力の区切りが 2 行の改行文字になる。レコード中のの改行の様子を簡易に調べる。
csv2tsv -~  ← 逆変換、つまり、TSV 形式のデータを CSV 形式に変換する。
csv2tsv --help  ← ヘルプ
csv2tsv --help opt ← オプションのヘルプのみ見る。opt は opti, optio, option, options でも良い。
```

4 その他

4.1 注意点となりうること (Perl に多少の経験がある人が気にするようなこと)

- **本モジュールが通常の CPAN にあるモジュールと異なる点:** 本モジュールは、単数または複数の Perl 言語で書かれたプログラムファイルが単独のファイルとして働くように作られている。各プログラムの中に、オンラインマニュアルヘルプも含まれているが、それらの文書は必ずしも POD 形式に従って書かれている訳ではない。通常のモジュールと異なり、ライブラリとなることを意図して作られていないので、本モジュールに含まれる *.pm ファイルを必要としているプログラムファイルは存在しない。
- **当該モジュールが別のモジュールを必要とするか:**
  - **本モジュールのインストール時** — Build の方法を使う場合は、Module::Starter を必要とするかも知れない。(本モジュールをダウンロードした場合に含まれていると考えられるが、場合による。) cpan がインストールされている状態で新たに cpanm をインストールする場合は、様々な方法があるが、cpan App::cpanminus を実行する方法もある。(この場合はつまり App::cpanminus に依存する。)
  - **プログラムの実行時** — 各プログラムファイルは、内部では use 文によりいくつかのモジュールに依存している。できるだけコアモジュールに依存するので、CPAN から新たなインストールを必要としないようにしてある。ただし List::MoreUtils と Text::CSV_XS と PerlIO::gzip などを必要により使うかもしれない。[*8]
- **利用者の計算機環境に影響を与えないか:**
  1. **インストール時に既存の実行可能なプログラムとのファイル名の衝突** — 本モジュールのプログラムファイルの名前と別のプログラムのファイルが衝突[*9]した場合に、実行時にどちらが実行されるかの問題、および、インストール時に上書きされる問題がある。cpan または cpanm の設定などに依存するが、/usr/local/bin/ に同名のファイルがあっても上書きがされないように、インストールするディレクトリを別の場所 (ローカルディレクトリなど) に変更することはできる。[*10]本モジュールのインストール方法に記載した cpan, cpanm, Module::Starter を使わない方法でインストールする場合には、パスの設定 (環境変数$PATH への書き込み) が通常は (特にその計算機に再びログインしてから本モジュールをすぐ使える状態にする場合は) 必要であるが、その場合でもそのパスに記載された複数のディレクトリの優先順位を検討する必要が発生しうる。[*11]
  2. **実行時の汚染チェック:** 一般に、Perl インタプリタの起動時に -T または -t のスイッチを与えることで汚染チェックが可能であるが、本モジュールにおいてそのような設定はまだ一部しか与えていない。
  3. **本モジュールにテストモジュールは付属しているか:** 未実装である。[*12]　実行結果が正しいかどう

[*8] **任意のモジュールがどの Perl のバージョンからコアモジュールとされているか知る方法:** たとえば List::Util が Perl 5.7 からコアモジュールとなっていることを知るには、次を実行する。

```
perl -MModule::CoreList -e 'print Module::CoreList->first_release('List::Util');'
```

[*9] ファイル名の衝突とは「一致」と同じ意味であるが、この一致は望ましいものではないので、あえて衝突と表記する。なお、ファイル名とは、パス名と混同されがちだがここでは異なる。パス名とは、ディレクトリ名が階層的にファイル名の前に/を区切り文字にして連結した文字列である。

[*10] ただし、その場合は cpan, cpanm, Module::Starter についての単なる利用者であること以上の知識が必要とされる。

[*11] echo $PATH | tr : "\n" や which -a コマンド名 のようなコマンドを使って検討することになるであろう。

[*12] 通常 Perl のテストはライブラリ (*.pm 形式のファイル) に対して行う。Test::More などのモジュールはそのように設計されて

かの検証には、下記の方法により確かめることが望ましい。

(a) 単純な例を入力して、動作を理解し、意図した通りに動作するか確認すること。

(b) 別のプログラムをうまく利用して異なる方法で結果をチェックをすること。

(c) Perl で書かれている本モジュールのプログラムファイルを閲覧して、意図しない動作が起こりにくいことを確認すること。(閲覧して、プログラムが十分にシンプルに書かれていることに疑問がある場合は、連絡して欲しい。たとえば、同じような動作を 2 箇所以上で実装していたりすると、バグの原因になりやすい。)

4.2 本モジュール CLI::LaTeX::Table の仕組み

- **どのバージョンの Perl に依存するか:**

  現状 5.14 以前に合わせている。(2018 年 4 月現在の Perl の最新バージョンは 5.26 である。) 世界のさまざまな機関で利用可能となるよう、2011 年以降の Perl であれば十分かもしれないという理由がある。技術的な要請として、乱数シードを設定する srand の動作が 5.14 以降で変更となったためでもある。[*13]それでも、出来るだけ古い計算機環境でも利用可能とすべく、5.1 でも動くように、case, say, state などの利用は出来るだけ避けるか避けようとしている。同様の理由で splice でのリファレンスの利用もしていない。

- **本モジュールに含まれるプログラムが どのモジュールをよくインポートしているか:**
  - `strict`
  - `warnings` → Perl 5.6 からコアモジュール。プラグマである。
  - **Getopt::Std** → オプションスイッチのオプションとそのパラメータを取得するため、および、--help で期待した動作をさせるため。Perl 5 の最初からコアモジュールである。
  - `FindBin` → 実行中のプログラムの名前を $Script で参照するため。Perl 5.3.7 からコアモジュール。
  - **Term::ANSIColor** → 出力に ANSI シーケンスカラーで色を付けるため。Perl 5.6 からコアモジュール。
  - `List::Util` → min や max 、sum 、sum0 関数を使う為。Perl 5.7.3 からコアモジュール。

いるように思われる (異なるかも知れないが、少なくともチュートリアルではそのようだ)。Perl のスクリプトに対して、汎用性の高いテストの方法を現在検討中である。

[*13] perldoc -f srand で参照が可能。