

学術研究のためのオープンソース・ソフトウェア(4)knitr

MIYAZAKI, Kenji / 宮崎, 憲治

(出版者 / Publisher)

法政大学経済学部学会

(雑誌名 / Journal or Publication Title)

経済志林 / The Hosei University Economic Review

(巻 / Volume)

83

(号 / Number)

4

(開始ページ / Start Page)

137

(終了ページ / End Page)

168

(発行年 / Year)

2016-03-30

(URL)

<https://doi.org/10.15002/00012867>

【研究ノート】

学術研究のためのオープンソース・ ソフトウェア (4) : knitr

宮 崎 憲 治

1 再現可能研究

再現可能研究 (reproducible research) が注目されている。研究成果をまとめた論文は読者が再現可能に記述する必要がある。つまり論文は、分析結果や図表がどのように作成されたかを再現可能のように記述しなければならない。また最近の学術雑誌ではデータ分析の際の現データやそのプログラミングコードの提出が求められている。

さらに再現可能研究においての論文は、現データからデータを読み込んで計算を実行し、図表を作成するプロセスを文書に埋め込むことができる論文を指すようになっている。そのように論文を作成しておけば、もとのデータが拡張されたり計算手法が修正されたときの計算結果の更新が楽になる。逆にそうしないとコピペミスなどが発生するおそれがある。つまり再現可能に学術論文をまとめることは、読者のためだけでなく、研究者自身のためにもなり、今後、身につけておかなければならない技術である。なお、この再現可能研究について、文芸的プログラミング (Literate programming) や動的文章 (Dynamic Documents) とも呼ばれている。ここでは、研究・プログラミング・文章のどれに重点を置いているかのニュアンス以外には区別せずに用いることにする。

歴史的に Knuth (1984) の文芸的プログラミングという言い方が古い。彼は $\text{T}_{\text{E}}\text{X}$ を考案している。プログラムの作成とその文書化を同時におこなうことによって、整合的な文章とプログラムを開発しようとするものである。クヌース自身、WEB というシステムを開発している。WEB はもともと $\text{T}_{\text{E}}\text{X}$ システムを記述するために開発された。現在、これを用いている人間はほとんどいない。このコンセプトをより単純化したものが、Ramsey (1994) の noweb である。

この noweb を統計ソフト S でもちいたのが Leisch (2002) の Sweave である。これはあとでフリーソフト R に拡張された。これによって $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ に R のコマンドを埋め込むことができる。そうすれば計算結果のコピペによるエラーなどが回避可能となる。Sweave は R に標準搭載されていて、パッケージのマニュアルはこれで作成されている。埋め込んだファイルの拡張子は S の場合 Snw で、R の場合 Rnw である。

Sweaveをよりカラフルに便利に用いたのが Xie の knitr である。当初 knitr は Sweave 互換であったが、現在は互換性はない。tex ファイルを出力するだけでなく、markdown で作成されたに R コードを埋め込むことによって、html にも出力することができる。さらに、python など、他の言語にも対応している。knitr のホームページは

<http://yihui.name/knitr/>

である。また彼自身が書いた書籍 Xie (2015) も参考になる。

本稿では knitrを紹介しつつ再現可能研究について紹介していく。knitr の導入の仕方は、R のコンソール上で

```
install.packages('knitr')
```

と入力すればよい。RStudio では自動的にインストールされる。RStudio については前回の研究ノート (宮崎, 2016) を参照されたい。

計量経済学において Koenker and Zeileis (2009) が再現可能性の研究に

について議論している。そこで議論しているのは R と Sweave であり, knitr については何も語っていない。また Gandrud (2015) は knitr を中心に再現可能論文の方法論について考察している。彼の考察は大いに参考になったが, 日本語での再現可能文献をどのようにすればよいかは言及していない。高橋 (2014) は knitr でどのように日本語を表記するかについて言及しているが, 本稿のように XeLaTeX についての議論はない。なお XeLaTeX については宮崎 (2015a) を参照されたい。

本論の構成は以下の通りである。まず Rnw ファイルについて説明し, Rmd ファイルの説明する。Rmd ファイルの説明の前に簡単にマークダウン記法についても触れている。次に Rmd ファイルのためのエディタについて私見を述べ, 最後に自分の使い方を示しながら研究ノートを終える。

2 Rnw

基本は, tex ファイルに次のようなコマンドを埋め込んで, 拡張子を Rnw とする。

```
<<(ラベル, )オプション>>=  
コマンド  
@
```

なおラベルは省略可能である。これをチャンクと呼んでいる。またインラインは `\Sexpr{...}` とすればよい。

たとえばデフォルトのもと

```
<<>>=  
rnorm(10)  
@
```

とすれば, コードとその結果が出力される (図 1)。もしコードの出力を抑

```

rnorm(10)

##[1]-0.1210453 -0.6899701 -0.9222474 -1.0707664 -11.6823629 1.0397053

##[7]-1.0465301 -0.6169041 -0.4400564 0.3833242

```

図1 knitrの出力例

えたいければ `echo=FALSE` を実施すればよい。

Rnwファイルの最初に

```

<<setup, include =FALSE, echo=FALSE, warning =FALSE, message
=FALSE>>=
library(knitr)
opts _chunk$set(echo =FALSE, warning=FALSE, message =FALSE)
@

```

とオプションとして指定しておけば、以降の他のチャンクでもこれが指定される。`warning=FALSE` や `message=FALSE` としないとライブラリの読み込みで意図しないメッセージが出力される。

RStudio 上で適切に設定していればワンクリックで、PDF を生成してくれる。R のコンソール上では

```

library(knitr)
knit('myfile.Rnw',encoding='utf-8')

```

とする。コマンドライン上では

```
Rscript -e "library(knitr); knit('myfile.Rnw', encoding='utf-8')"
```

とする。そうして tex ファイルを生成する。その後、通常の latex を実行する。日本語まじりの文章でも対応しているが、platex は RStudio 上では使えない。

逆に Rnw ファイルから R ファイルを作るには、R のコンソール上で

```
purl("myfile.Rnw")
```

を実行すればよい。

2.1 図

knitr で図を作成するには、別途ファイルを作成して保存して、それを読み込むやり方と直接表示させるやり方がある。R で図の作成方法については宮崎 (2016) を参照されたい。一旦保存するやり方は前の研究ノートで示した。作図についてのチャンクを付け加えれば良い。直接表示するには次のようにする。

```
\documentclass{bxjsarticle}
\usepackage{xltextra}
\usepackage{zxjatype}
\usepackage[ipa]{zxjafont}
\usepackage{graphicx}
\begin{document}
\begin{figure}[ht]
\centering \caption{figtitle}\label{fig1}
<<echo =FALSE, fig.width=7, fig.height=5, warning =FALSE,
message =FALSE, dev ='pdf'>>=
curve(dnorm(x), xlim =c( -3,3))
@
\end {figure}
\end {document}
```

`dev` の指定がないと `dev='png'` と `png`形式の図となる。図の指定単位はインチである。また作成した図は特に指定がない限り同じフォルダにサブフォルダ `figure` が作られそこに保存される。

日本語のフォントを用いる場合、デバイスを `cairo_pdf` に指定してフォントを設定する。たとえば

```
<<echo=FALSE, fig.width=7, fig.height=5, warning=FALSE,
message=FALSE, dev='cairo_pdf'>>=
par(family="IPAGothic")
curve(dnorm(x), xlim=c(-3,3), main="正規分布")
@
```

とする。ここでは `par(family="IPAGothic")` としている。なお日本語まじりだと、作成に時間がかかる。また、警告がでるのでオプションの `warning` を必ず `FALSE` にしなければならない。

`ggplot2` を用いるなら、デバイスに `cairo_pdf` を指定するのは同じであるが、フォントの指定の仕方が異なる。たとえば

```
\begin{figure}[ht]
\centering\caption{figtitle}\label{fig1}
<<echo=FALSE, fig.width=7, fig.height=5, warning=FALSE,
message=FALSE, dev='cairo_pdf'>>=
library(ggplot2)
theme_set(theme_gray(base_family="IPAGothic"))
qplot(cars$speed, cars$dist, main="散布図")
@
\end{figure}
```

とする。ここでは `theme_set(theme_gray(base_family="IPAGothic"))` と変更していることに注意されたい。

`tikzDevice` を用いるなら、以下のようにする。

```
\documentclass{bxjsarticle}
\usepackage{xltextra}
\usepackage{zxjatype}
\usepackage[ipa]{zxjafont}
\usepackage{tikz}
\begin{document}
\begin{figure}
<<echo=FALSE, fig.width=7, fig.height=5, warning=FALSE,
message=FALSE, dev='tikz'>>=
library(tikzDevice)
options(tikzDefaultEngine='xetex')
curve(dnorm(x), xlim=c(-3,3), main="正 規 分 布 :  $N(\mu, \sigma^2)$ ")
@
\end{figure}
\end{document}
```

本文のほうに `\usepackage{tikz}` を忘れずにつけておかなければならない。

2.2 表

図と同じように別途作成して保存するか、直接表示するかのやり方がある。R での作表方法については宮崎 (2016) を参照されたい。図と違って直接表示した場合ファイルが作成されない。このとき次のように `results='asis'` を付け加える。

```
\documentclass{bxjsarticle}
\usepackage{xltextra}
\usepackage{zxjatype}
\usepackage[ipa]{zxjafont}
```

```

\begin{document}
<<echo=FALSE, results='asis'>>=
data(cars)
fm<-lm(dist~speed, data=cars)
library(xtable)
print(xtable(fm),floating=FALSE, comment=FALSE)
\end{document}

```

下から二行目で `comment=FALSE` としているのは、コメントを表示しないためである。Rnw 形式ではこれを付け加えなくても動作するが、後で説明する Rmd形式の出力の際にはこのコメントが必要である。また `floating=FALSE` としているのはフロート体の指定を自分でするからである。これがなければ table 環境の指定になる。さらにキャプションやラベルの指定もできる。私は自分で指定するようにしている。

フロート体で出力するには次のようにする。

```

\documentclass{bxjsarticle}
\usepackage{xltextra}
\usepackage{zxjattype}
\usepackage[ipa]{zxjafont}
\begin{document}
\begin{table}[ht]
\centering\caption{tabtitle}\label{tab1}
<<echo=FALSE, results='asis'>>=
...
...
@
\end{table}
\end{document}

```

フロート体の指定を自分ですれば `threeparttable` を用いて注釈付の図が作図可能になる。

```
\begin{table}[htb]
\centering\caption{tabtitle}\label{tab1}
\begin{threeparttable}
<<echo=FALSE, results='asis'>>=
...
...
@
\begin{tablenotes}\footnotesize
\item[1] Note. ..
\item[2] Souce
\end{tablenotes}
\end{threeparttable}
\end{table}
```

`xtable` 以外の作表について述べる。`texreg` の場合、次のようにする。

```
texreg(list(fm1, fm2, fm3), table = FALSE)
```

フロート体を未指定にしている。また `stargazer` の場合、次のようにする。

```
stargazer(fm0, fm1, header=FALSE, float=FALSE)
```

フロート体の未指定だけでなく、Rmd 形式で出力する際にコメント出力しないように `header=FALSE` を付け加えている。

2.3 beamer

beamer に対しても適応可能である。beamer については前回の研究ノート（宮崎，2015a）を参照されたい。

```

\documentclass{beamer}
\usepackage{xltextra}
\usepackage{zxjatype}
\usepackage[ipa]{zxjafont}
\usetheme{Madrid}
\usecolortheme{beaver}
\usefonttheme{professionalfonts}

\title{Test framebreaks}
\author{宮崎憲治}
\begin{document}
\maketitle
<<setup, include=FALSE>>=
library(knitr)
@
\begin{frame}[t, shrink, fragile]
\frametitle{分析結果}
<<size="footnotesize",out.width=".7\\linewidth">>=
data(cars)
fm<-lm(dist~speed, cars)
summary(fm)
@
\end{frame}
\begin{frame}
\frametitle{作図}
<<echo=FALSE, fig.width=7, fig.height=5>>=
data(cars)
fm<-lm(dist~speed, data=cars)
plot(dist~speed, data=cars)
abline(fm)
@
\end{frame}
\end{document}

```

フレームオプションで入力や結果コマンドが含まれる場合は、`fragile` を付け加えなければならない。また、結果が長いとファイルに収まらないことが多々ある。チャンクオプションで、`size="footnotesize"` とするとうまくいくことが多い。

フレームオプションで `shrink` を付け加えると、自動的に大きさを調節してくれる。自動的な調整に満足が行かない場合、`shrink=20` のように数字を変えるとうまくいく。なお、数ページにわたる結果を `shrink` で表示するときは、非常に小さくなる。数ページにわたる結果をださないようにするか、分離して表示したほうがよい。

分離して表示するには、`allowframebreaks` というオプションをつけるとうよい。ただ `beamer` での `fragile` 環境の出力は `kframe` からはみ出た部分はそのまま無視してしまう。これを防ぐには最初のところに以下を入力する。

```
<<setup, include=FALSE>>=
library(knitr)
# the kframe environment does not work with allowframebreaks,
so remove it
knit_hooks$set(document = function(x) {
  gsub('\\\\\\(begin|end)\\\\{kframe\\\\}', '', x)
})
@
```

なお `default` 以外のテーマの場合、`\frametitle` をなしにしないといけない。

2.4 bibtex

参考文献も扱うことができる。参考文献については宮崎（2015b）を参照されたい。ただ RStudio 上から `upbibtex` を扱うことができないため、

日本語の参考文献がある場合は、扱いに気をつけないといけない。すでに `upbibtex` で `bb1` ファイルが生成されていて変更していないのなら、RStudio 上から実施すればよい。そうでないと RStudio の外からコンパイルして、`bb1` ファイルを生成する必要がある。

3 md (マークダウン)

`knitr` は `Rmd` という形式でも作成することが可能である。`Rmd` は `R+md` のことで `R` を埋め込んだ `md` 形式のことである。マークダウン記法とは、軽量マークアップ言語の一つである。ジョン・グルーバー (John Gruber) によって2004年に開発され、多くの開発者の手を経ながら発展してきた。通常の `html` でマークアップを入力したときにタグ打ちが面倒である。このマークダウン記法をつかえば、簡単なルールをまもれば簡単に簡単に作成でき、`html` に出力可能である。GitHub などのサイトでは `html` に変換せずにマークダウン形式のままでファイルをアップロードすればサイトのほうで変換してくれる。`html` 以外にも `pandoc` を用いて様々な形式に変換してくれる。`pandoc` はマークダウンだけでなく様々なファイル形式を互いに変換してくれるツールである。公式サイトは以下にある。

<http://pandoc.org/>

このマークダウン記法を `R` に埋め込んだ形式を `Rmd` 形式という。RStudio をつかえば、コマンド部分を `knitr` で出力し、`pdf` だけでなく、`html` や `word` 形式でもファイルを作成することができる。なお `pandoc` は RStudio をインストールすれば自動的に導入される。`R` の埋め込み方については次節で解説し、以下この節ではマークダウン記法の基本を概説する。

3.1 タイトル

RStudio で `html` 形式で新規作成した場合、

```
---  
title: "Untitled"  
author: "Kenji Miyazaki"  
date: "2014年12月31日"  
output: html_document  
---
```

と出力される。タイトルと著者名と日付と出力形式が指定されているが、あとで変更可能である。

3.2 文章構造

セクションは行頭に # を入れることによって作成する。その数によって構造が変わる。

```
# This is H1  
## This is H2  
### This is H3  
#### This is H4  
##### This is H5  
##### This is H6
```

最初の2つは次でも代替可能である。

```
This is H1  
=====  
This is H2  
-----
```

3.3 文字修飾

- イタリックは `*...*` もしくは `_..._` で囲む。

- 太字は `**...**` もしくは `__...__` で囲む。
- 取消線をつけるには, `~~...~~` で囲む。

3.4 箇条書き

- 番号なしリストは行頭に `*`, `+`, `-\` のいずれかを置く。
- 番号付きリストは行頭に数字に続けてピリオドを置く。
- 階層リストを作成するには, 行頭に 4 つの空白または 1 つのタブを置く。

3.5 引用

先頭に `>` で引用ブロックとなる。

3.6 コード

- インラインコードはシングलバッククオート (```) で囲む。
- コードブロックはバッククオートを 3 つ並べた行 (`````) で挟む。

3.7 数式

- インライン数式モードは `$...$` もしくは `\(...\)` で囲む。
- ディスプレイ数式モードは `$$...$$` もしくは `\[...\]` で挟む。
- 式番号は自動的につかないので `\tag{...}` を用いる。
- 複数の数式の場合, `aligned` 環境を使うとよい。

```

$$
\begin{aligned}
B' &= -\partial \times E, \tag{*} \\
E' &= \partial \times B - 4\pi j, \\
\end{aligned}
$$

```

3.8 作表

表は次のように作表する。

First Header	Second Header	Third Header
Content Cell	Content Cell	Content Cell
Content Cell	Content Cell	Content Cell

もし配列を調整したい場合、次のようにする。

First Header	Second Header	Third Header
Left	Center	Right
Left	Center	Right

3.9 作図

- ウェブ上の画像は `![テキスト](http://www.dot.com/picture.png)` とする。
- 相対パス上の画像は `![テキスト](figures/picture.png)` とする。

3.10 リンク・相互参照

- email リンクは `<example@example.com>` とする。
- インラインリンクは同様に `<http://chenluois.com>` とする。
- リンク先を隠すのなら `[Smaller](http://25.io/smaller/)` とする。タイトル付きにしたければ `[Resize](http://resizesafari.com "a Safari extension")` とする。

なお、相互参照は pdf 出力にかんしては latex のコマンドをそのまま入力しておくとも pdf 出力について出来るようである。html 出力や docx 出力

は基本的にできない。(http://stackoverflow.com/questions/ 13848137/figure-captions-references-using-knitr-and-markdown-to-html)

3.11 参考文献

参考文献は `pandoc` を導入することで `latex` と `html` 出力について出来るようである。ただ `bibtex` エンジンを `upbibtex` に指定できないため、日本語文献の参照は現状では厳しい。

3.12 脚注

- 脚注を入れるには、本文中に脚注を入れたい箇所に `^[脚注内容]` とする。
- もしくは脚注を入れない箇所に `[^id1]` と書き。別の場所に、

<code>[^id1]:脚注内容</code>

とする。いずれの方法でも、自動的に番号が振られ、脚注内容が文章の最後に置かれる。

3.13 その他

- 行末に 2 個以上の半角スペースを入れると、そこで改行される。
- 行頭から行末まで 3 個以上のアスタリスク (*) もしくはハイフン (-) があり、前後の行が空白行の場合、水平線に変換される。

4 Rmd

この節ではマークダウン記法にどのように R コードに埋め込んで、RStudio で文書を作成していくかみていく。Rnw 形式は pdf しか出力できなかったが、html や word 形式でもファイルを作成することができる。より詳しくは以下を参照されたい。

<http://rmarkdown.rstudio.com/>

また

<http://www.rstudio.com/wp-content/uploads/2015/03/rmarkdown-reference.pdf>

にあるチートシートも有益である。

4.1 原則

原則としては

```
```${r オプション}
コマンド
```
```

を付け加える。インラインは`r...`とすればよい。pdf 出力の際のオプションは Rnw のときと同じである。RStudio で R Markdown 形式のファイルを新規作成すると簡単なテンプレートが出てくる。html 出力はブラウザがそのまま出力することができる。word 出力のためには Microsoft word がインストールされていないといけない。pdf 出力は latex がインストールされていないといけない。latex として texlive 2013以上が必要である。html 出力と word 出力はデフォルトのままで日本語まじりの文章を出力することができる。pdf 出力の場合、後述するように日本語出力のためには少し工夫が必要である。

さらにプレゼンテーションファイルも簡単に作ることができる。pdf 出力は beamer 形式の出力が可能である。そのやり方は後述する。html 出力では2種類のスタイルがある。ひとつは **isoslides** 形式で非常にグラフィカルにあらわせる。ただスライドに収まらない場合、表示されなくなるので、区切り方に注意する必要がある。もう一方は **slidify** である。これは派手なグラフィクスはないが、スライドにおさまらない場合は単に下にス

クロールするだけでよいので使い勝手がよい。

作り方は最初のテンプレートでえらべば例付きで新規ファイルを開いてくれるので、それをもとに修正すればよい。基本は **##** で表題となり、箇条書きは以下のようにする。

```
## Slide with Bullets
```

```
-Bullet 1
```

```
-Bullet 2
```

```
-Bullet 3
```

R の実例付きのスライドを作成したい場合、Rmd を使えば手早く資料が作成できる。注意することは私の大学のようにプロジェクターの解像度がそれほど細かくない場合、大きすぎて不格好になる。こうしたときは、単に html ファイルをそのまま作り、適宜拡大しておくとうまいだろう。

また Shinyというアニメーションを使い、動きのある html ファイルを作ることができる。ここでは紹介しないが以下で参照していただきたい。

<http://shiny.rstudio.com/>

4.2 pdf出力の注意

しかし、pdf 出力の場合、たとえインストールされていても日本語まじりの文章をデフォルトのままでは出力できない。

もし 2015年 9月にバージョンアップした BXjscls パッケージの BXJS文書クラスに更新しているなら、ヘッダーはつぎのようにする。

```
---
```

```
title: "タイトル"
```

```
author: "宮崎憲治"
```

```
output:
  pdf_document:
    latex_engine: xelatex
documentclass: bxjsarticle
classoption: xelatex,ja=standard
geometry: no
---
```

BXjscls が内部的に使うから `geometry` を使用してはいけない。

<http://qiita.com/naqtn/items/7469a07ea797df5e2ecc>

なお、ここでは `xelatex` であるが、最新の `BXjscls` を使っていれば、`latex` エンジンでも `pdflatex` に指定しても、`lualatex` に指定しても動作する。残念ながら `platex` はうまくいかない。2つのエンジンについての表記の違いは以下を参照されたい。

<https://oku.edu.mie-u.ac.jp/~okumura/texwiki/?BXjscls>

もし最新の `BXJS` クラスに更新していないのなら、`latex` エンジンを `xelatex` にして、次のようにする。

```
---
title: "タイトル"
author: "宮崎憲治"
header-includes:
  -\usepackage{xltextra}
  -\usepackage{zxjatype}
  -\usepackage[ipa]{zxjafont}
output:
  pdf_document:
    latex_engine: xelatex
```

```
documentclass: bxjsarticle
geometry: no
---
```

日本語を使用するためのヘッダーを挿入している。ここでも **geometry** を使用してはいけない。

beamer はで日本語のまじり文章を PDF 作成する場合, latex エンジン を xelatexにするならヘッダーをつぎのようにすれば出力可能である。

```
---
title: "タイトル"
author: "宮崎憲治"
header-includes:
  -\usepackage{xltextra}
  -\usepackage{zxjatype}
  -\usepackage[ipa]{zxjafont}
output:
  beamer_presentation:
    pandoc_args: --latex-engine=xelatex
---
```

更に、わたしは Rmd ファイルの最初に次のセットアップを加えている。

```
```{r setup, include=FALSE}
library(knitr)
opts_chunk$set(size = 'footnotesize')
options(width=60)
knit_hooks$set(document = function(x) {
```

```
gsub('\\\\\\(begin|end)\\\\{kframe\\\\}', '', x)
})
...
```

デフォルトのままだと文字幅がはみ出してしまうので、フォント小さめにしている。Rnw と同様に複数にスライドを分けて出力されるための設定を加えている。

この設定を加え、さらに分割表示される場合には以下のように `{.allowframebreaks}` を付け加えるよい。

たとえば

```
test {.allowframebreaks}
``{r}
example(step, run.donttest=TRUE)
...
```

とすると、そのフレームの latex のソースコードは以下のようになる。

```
\begin{frame}[fragile,allowframebreaks]{test}
\begin{Shaded}
\begin{Highlighting}[]
\KeywordTok{example}\NormalTok{((step, }\DataTypeTok{run.
donttest =} \OtherTok{TRUE}\NormalTok{)})}
\end{Highlighting}
\end{Shaded}
....
```

### 4.3 図の注意

図は Rnw とほぼ同じように出力することができる。latex の場合 pdf 出

力でよいだろう。word や html だと png 形式での出力のほうが望ましい。チャンクにオプションつけなければ、`dev='png'` を意味し、png 形式のファイルが作成される。

なお Rmd 形式において `tikz` で日本語が含まれているときには以下のように変更しなければならない。

```
```\{r,dev='tikz', warning=FALSE, message=FALSE, echo=FALSE}
library(tikzDevice)
options(tikzDefaultEngine='xetex')
options(tikzXelatexPackages = c(getOption('tikzXelatexPackages'),
'\\usepackage[ipa]{zxjafont}'))
curve(dnorm(x), xlim=c(-3,3), main="正規分布:  $N(\\mu, \\sigma^2)$ ")
```
```

Rnwファイルと違い、オプション `tikzXelatexPackages` をつけなければならない。ただ `lattice` だと日本語がうまく表示されない。

表は latex と html は同じように出力することはできない。word については次小節で説明する。latex の場合、Rnw のときと同様である。もしフロー体を Rmd でおこなうには、引数を調整すればよい。ただ図に番号が付くが、それを参照することを Rmd 形式では簡単にできない。また不要なコメントが出力されるのでコメントアウトしなければならない。それらの指定はすでに説明した。

htmlの場合、`xtable` では次のようにオプション `type='html'` を付け加えればよい。

```
```\{r echo=FALSE, results='asis'}
library(xtable)
print(xtable(summary),type="html")
```
```

stargazer では次のようにオプション `type= 'html'` を付け加えればよい。

```
```${r echo=FALSE, results='asis'}
library(stargazer)
stargazer(fm1, fm2, type="html")
```
```

texreg では次のようにコマンド `htmlreg` に変更すればよい。

```
```${r echo=FALSE, results='asis'}
library(texreg)
htmlreg(list(fm1, fm2))
```
```

#### 4.4 word の注意

word の場合は表をうまく扱えない。kable を使えばマークダウン形式の表を出力してくれるので、word でも pdf でも html でもあつかえる。

```
```${r echo=FALSE, results='asis'}
library(knitr)
library(AER)
fm<-lm(dist~speed, data=cars)
hc <-vcovHC(fm, type="HC1")
ct <-coeftest(fm, vcov = hc)
kable(as.matrix(ct[,]), format="markdown")
```
```

kable は knitr を導入することで実行できるようになる。ただ kable はオブジェクトとして dataframe と matrix しか扱えない。もうすこし表を装飾したい場合 pander というライブラリを用いればよい (<http://rapporter>).

github.io/pander/)。なお `pander` も `summary(fm)` は上手く出力するが、`coeftest(fm)` はやはり上記のような修正が必要である。

`word` の場合、`stargazer` や `texreg` などの複数の回帰分析結果をまとめる表は一発で綺麗に変換してくれない。対応策として、綺麗さを犠牲にするならコマンド `screenreg` でテキスト表示するようにすればよい。`screenreg` のとき、チャンクオプションの `results` が必要なくなることに注意されたい。つまり `stargazer` なら

```
``{r echo=FALSE}
library(stargazer)
stargazer(fm1, fm2, type="text")
``
```

とし、`texreg` なら

```
``{r echo=FALSE}
library(texreg)
screenreg(list(fm1, fm2))
``
```

とする。

なお一発変換を犠牲にするなら `html` 形式で出力してそれをコピペすればよい。もし `texreg` を使うなら外部ファイルに保存できる。例えば以下のようにすればよい。

```
htmlreg(list(fm1, fm2), file="mytable.doc", inline.css= FALSE,
doctype=TRUE, html.tag=TRUE, head.tag=TRUE, body.tag=TRUE)
```

また次のようにすれば、ひとつのファイルで3つのファイル形式を切り替えて表を出力することができる。

```

title: "3フォーマット対応文書"
documentclass: bxjsarticle
geometry: no
output:
 word_document: default
 pdf_document:
 latex_engine: xelatex
 html_document: default
classoption: xelatex, ja=standard

```{r, echo=FALSE}
library(knitr)
out_type <- opts_knit$get("rmarkdown.pandoc.to")
```

データ `cars` を用いて回帰分析を実行する。
```{r}
fm0<-lm(dist~speed, data=cars)
fm1<-lm(dist~speed + I(speed^2), data=cars)
```

結果は以下のとおりである。
```{r, message=FALSE,echo=FALSE}
library(stargazer)
if (out_type == "docx") {

  stargazer(fm0, fm1, type="text")
}
```
```

```

```{r, results='asis',echo=FALSE}
switch(out_type,
  html = stargazer(fm0,fm1, type="html"),
  latex = stargazer(fm0,fm1, header=FALSE, float=FALSE)
)
```

```

## 5 その他

ここでは knitr についてのいくつかの関連トピックスについて述べていく。

### 5.1 外部エディタ

RStudio はあまり日本語入力が得意ではない。使っている OS によっては、日本語入力中に入力状況が全く表示されない場合がある。日本語入力に不満があるなら、別のエディタで Rmd ファイルを編集すればよい。RStudio でそのファイルを開いたままにしておいて、外部エディタの編集結果を保存する。RStudio のソース・ウインドウを選択すると編集結果が自動的に反映される。RStudio 上ではコードの実行や knitr を実施に専念すればよい。

Rnw ファイルの編集には tex ファイルの編集に使ったエディタを用いればよい。外部エディタについて

<http://yihui.name/knitr/demo/editors/>

に色々書かれている。ここでの設定だと日本語での実行ができない。以下 Texmaker で使用するための設定を記述する。Texmaker については宮崎 (2015a) を参照されたい。

Texmaker ではまず [User]=>[User Commands]=>[Edit User Commands] を実施して, [Menu Item] を knitr とする。次に [Command] 欄に, mac の場合,

```
"/usr/local/bin/Rscript" -e "knitr::knit2pdf('%Rnw',
compiler='xelatex')"
```

と入力すればよい。windows の場合,

```
"C:/Program Files/R/R-*.*/bin/Rscript.exe" -e "knitr::
knit2pdf('%Rnw', compiler = 'xelatex')"
```

にする。

Rmd ファイルの編集の場合, マークダウン形式に対応したエディタが多数ある。たとえば Emacs, sublime text, Atom などである。私は Emacs の markdown-mode+polymode+ess を組合せて使っている。ただ Emacs は20年以上前の古いエディタなのでこれからはじめる人におすすめできない。私見であるが Atom (<https://atom.io/>) が今後主流になりそうである。

## 5.2 外部ファイル

Rnw ファイルや Rmd ファイルに R コードを埋め込まず, 外部ファイルを読み込むことができる。詳しくは以下のサイトを参考されたい。

<http://yihui.name/knitr/demo/externalization/>

例えば hoge.R という名前の以下の外部ファイルがあるとする。

```
----test-a ----

1+1

x = rnorm(10)

----test-b ----
```

```
mean(x)
sd(x)
```

ここで、ラベルとして、**## ----ラベル名----** とする  
これを読み込ませるには、まず、

```
<<external-code, cache=FALSE, echo=FALSE>>=
read_chunk('hoge.R')
@
```

とすればよい。チャンクオプションに **cache=FALSE** と指定しなければならない。**echo=FALSE** とすればインプットを隠すことができる。  
そうして、本文中に、

```
<<test-a>>=
@
```

とすればよい。

### 5.3 spin & stitch

RStudio で R ファイルから、'compile notebook' ボタンをおせば、html ファイルなどを生成してくれる。これは knitr の **spin** というコマンドを動かしている。

R ファイルにおいて、**#'** で始まるものが R マークダウンファイルでのテキスト行に読み替えられる。また **#+** ではじまるものがチャンクオプションに変換される。

たとえば以下のファイル **test.R** があるとする。

```
#' ---
#' title: "Sample Document"
```

```
#' ---

#' # carsのプロット
#' #' carsをプロットします。

#+ echo = FALSE
#通常のコメントはコメントとして解釈されます。
plot(cars)

#' #インラインコードの書き方です。#' carsのサイズは
#{{dim(cars)}}
#'です。
```

このとき以下のコマンドを実行する。

```
spin('test.R', FALSE)
```

そうすると以下のファイル `test.Rmd` が生成される。

```

title: "Sample Document"

carsのプロット
carsをプロットします。

``{r echo = FALSE}
#通常のコメントはコメントとして解釈されます。
plot(cars)
```

#インラインコードの書き方です。
```

```
carsのサイズは  
`r dim(cars)`  
です。
```

これより、各種ファイルへ変換することができる。もしくは直接 RStudio の「compile notebook」ボタンをクリックすればよい。

R ファイルからドキュメントを作成数コマンドとして、`spin()` のほかに、`stitch()` がある。これは `spin()` と違い、予めテンプレートを作成してそれをもとにドキュメントを作成する。詳しくはヘルプを参照されたい。

まとめ

この研究ノートでは再現能研究のための R のパッケージを紹介した。tex ファイルに R コードを埋め込んだ Rnw ファイルだけでなく、マークダウン記法に R コードを埋め込んだ Rmd ファイルについて、どのようにすればよいかを概説した。とくに Rmd ファイルでは図表を pdf 形式だけでなく html 形式や docx 形式での出力が可能になることを見てきた。

研究活動において研究ノートと論文は同じではない。研究ノートは幾つかの試行錯誤を網羅的にまとめたものである。一方、実際の投稿論文では研究ノートの一部となることが多い。Rmd ファイルは扱い易いが、参考文献や式番号付きの数式表示などでいろいろ面倒である。そのため研究ノートは Rmd ファイルで、実際の論文は Rnw ファイルにしておいたほうが良いだろう。ただ Rmd ファイルの計算手順をコピーするとミスが起こりうるので避けなければならない。

そのため、次のように2つのファイルを使い分ければよいだろう。すべての計算結果やメモを Rmd ファイルで日本語で記録する。たとえば `hoge.Rmd` とする。これをもとに html ファイルを生成し、中間報告やすべての計算を実施する。すべてのチャンクにラベルを付けておく。

そうして, Rmd ファイルから R の実行コマンドを抽出する。具体的にコンソール上で以下を実行する。

```
purl("hoge.Rmd")
```

そうして生成したファイル **hoge.R** を外部ファイルとする。英語もしくは日本語で Rnw の論文を作成して、論文に必要な実行結果などを適宜挿入する。そうすれば、再現可能な論文が作成できる。別途学会報告用に英語や日本語のプレゼンファイルも必要であろう。この際は表現のコピペは仕方ないにせよ、計算過程はなるべく単一で実施すべきである。プログラムの世界で知られている DRY (Don't Repeat Yourself) 原則 (ハント・デビッド, 2000) は再現可能研究には不可欠である。

実は私は論文作成のとき Rnw ファイルをほとんど扱わない。さらに latex も共同研究者が latex しか使えないときにしか latex を扱わない。実はそれをグラフィカルに表示してくれる LyX というアプリケーションで多くの作業を実施している。LyX は knitr と連動して外部ファイルを読み込んでくれる。LyX については次回紹介していきたい。

参考文献

- Gandrud, Christopher (2015) *Reproducible Research with R and R Studio*, Boca Raton: Chapman and Hall/CRC, 2nd edition.
- Knuth, D. E. (1984) “Literate Programming,” *The Computer Journal*, Vol. 27, No. 2, pp. 97–111, February, DOI: <http://dx.doi.org/10.1093/comjnl/27.2.97>.
- Koenker, Roger and Achim Zeileis (2009) “On reproducible econometric research,” *Journal of Applied Econometrics*, Vol. 24, No. 5, pp. 833–847, August, DOI: <http://dx.doi.org/10.1002/jae.1083>.
- Leisch, Friedrich (2002) “Sweave: Dynamic Generation of Statistical Reports Using Literate Data Analysis,” in Härdle, Wolfgang and Bernd Röß eds. *Compstat*, Heidelberg: Physica-Verlag HD, pp. 575–580.
- Ramsey, Norman (1994) “Literate programming simplified,” *IEEE software*, No. 5, pp. 97–105.
- Xie, Yihui (2015) *Dynamic Documents with R and knitr*, Boca Raton: Chapman and Hall/CRC, 2nd edition.
- 高橋康介 (2014) 『ドキュメント・プレゼンテーション生成』, 共立出版, 東京.
- ハント, A・T.デビッド (2000) 『達人プログラマー—システム開発の職人から名匠への道』, ピアソンエデュケーション, (村上雅章訳).
- 宮崎憲治 (2015a) 「学術研究のためのオープンソース・ソフトウェア (1) : XeLaTeX」, 『経済志林』, 第 82 巻, 第 4 号, 285–321 頁, 3 月.
- (2015b) 「学術研究のためのオープンソース・ソフトウェア (2) : BiBTeXと Zotero」, 『経済志林』, 第 83 巻, 第 2 号, 119–149 頁, 3 月.
- (2016) 「学術研究のためのオープンソース・ソフトウェア (3) : Rと RStudio」, 『経済志林』, 第 83 巻, 第 3 号, 3 月.