Converting LaTeX Legacy R Journal Articles into R Markdown Articles using texor and rebib

by Abhishek Ulayil, Mitch O'Hara-Wild, Christophe Dervieux, Heather Turner, and Dianne Cook

Abstract An abstract of less than 150 words.

1 Introduction

The R Journal is the primary open-access outlet for publications produced by the R community. It was born in 2008, evolving from a newsletter, that ran from 2001, into a more formal article publication to encourage documenting statistical computing research.

The format is constantly evolving. Early articles were typeset using LateX (The LaTeX Project 2023), from a specific, but changing, template. This requires that code is separated from the documentation, and there is a chance that code chunks in the paper don't reproduce the results reported. With the emergence of dynamic document systems like R Markdown (Xie, Allaire, and Grolemund 2018) a tight-coupling of code and documentation is possible. Code chunks are dynamically executed when the document is typeset using a system like **knitr** (Xie 2015), making reporting of computing research more reproducible.

In 2019, with the help of funding from the R Consortium it was decided that it was time to update operations. One aspect of this was to change from LaTeX paper submissions to a more reproducible format, where code was embedded in the document, and the output could be both HTML and pdf. There are numerous benefits of HTML format:

1. Articles can include interactive graphics and tables.
2. The format is more accessible to screen readers making the work more accessible to vision-impaired researchers.

This latter point is a reason to consider converting all of the legacy articles into HTML.

A key decision for creating conversion software was to decide to directly convert LaTeX to HTML, or PDF to HTML, or LaTeX to R Markdown, and then use the current journal tools to create the HTML. The latter approach was decided to be the most versatile and useful. If an article can be converted from LaTeX to R Markdown, it would help authors make the transition to reproducible publishing, beyond what the R Journal needed. Once an article is in R Markdown format it can be adapted to include the code for dynamic execution.

In addition to article format, changes to the web site structure were important for delivering the publication. Web site architectures are also constantly evolving, and the emergence of **distill** (Dervieux et al. 2022) allows for the journal web site to optimally deliver R Markdown articles.

The **rjtools** was developed to create articles using R Markdown for the R Journal, and to embed them into the journal web site. The packages described here, **texor** and **rebib** describes software to convert legacy LaTeX format articles into Rmarkdown, so that they can be rendered in HTML in the new web site.

The paper is organised as follows. Section XXX gives an overview of the conversion process. Section XXX describes examples of pre-processing using regular expressions, and section XXX provides examples of post-processing using lua filters. Section XXX describes tools to do special handling of bibliography files. Section XXX runs through an example conversion process.

2 The internals of converting from LaTeX to R Markdown

The decision to convert to R Markdown format means that the final output to pdf and HTML will depend on Pandoc (MacFarlane 2023). Pandoc is a versatile document conversion program written in Haskell that is core to numerous documentation systems, including R Markdown and Quarto. Pandoc first converts a document into an abstract syntax tree. From this, it can convert to a different format, including custom ones.

Pandoc can be used to do the conversion from LaTeX to R Markdown also. However, additional pre-processing needs to be done to handle special R Journal LaTeX styling. And further post-processing

needs to be done to handle specific R Journal R Markdown styling. The **texor** package contains functionality to handle this pre- and postprocessing of the document.

DC: Add a diagram here showing flow of work, just pre-process .tex -> pandoc -> lua ; rebib operations???

The supplementary materials has example LaTeX documents that allow the reader to see how different common patterns in the legacy documents are handled with the conversion. These include:

- `code-env`: Explains how different code environments defined by the R Journal style are handled, and additional details such as code in figure environments, and code in table environments.
- `math-env`: Examples of inline math, display math, how equation numbering is handled by a Lua filter to convert from LaTeX labelling to R Markdown labelling.
- `figure-env`: Explains how the variety of figure definitions are handled in the conversion, including different image formats, numbering, captions, labelling, multiple images, and `tikz` (Josh Cassidy 2013) images.
- `table-env`: Examples of how a variety of table types are converted, including multicolumn, complex and wide tables.
- `lua-filters`: Overview and lots of small examples of Lua filters to handle the custom output needed for the R Markdown format.
- `metadata`: This has a collection of additional format handling including extracting metadata like author names and affiliations, article identifiers used in the review process, and handling citations, footnotes and links.
- `bibliography`: The bibliography was handled differently over the years of the journal, and this details how to use the `rebib` functionality to handle `bbl` files, embedded `bbl`, to convert into the standard `.bib` format.

The explanations in the next sections are extracted from these examples. In each of these folders there is a `RJwrapper.tex`, and `.tex` file, with the extra template files `RJournal.sty` and `Rlogo-5.png`. These match the legacy template file structure, from which the `RJwrapper.pdf` file is created. If you follow the instructions in the `README.md` you can create the R Markdown and HTML versions of the document.

Pre-processing using regular expressions

LaTeX is very descriptive language, that allows authors substantial freedom for customization. Markdown (Gruber 2002), on which R Markdown is based, is more restrictive and was born to make it easier to create web pages without the distraction of a gazillion HTML tags. The beauty of Markdown is that it allows the author to focus on writing, without format cluttering the text. The drawback is that it is simple typesetting, optimized for web delivery.

While pandoc can do most of the heavy-lifting, it cannot cope with all the freedom with which LaTeX documents are written. An example of this is with formatting of code. Pandoc only handles the `verbatim` environment, but there are many ways to format code in LaTeX, and the R Journal template has a special `\code{}` command. If the code environment is not verbatim, then pandoc will also try to process the actual code content as LaTeX commands and will likely lose details. It is better to convert these synonyms into `verbatim` environments this prior to passing the document to pandoc.

The functions in **texor** that handle the pre-processing using regular expressions are:

- `stream_editor()`: operates like the `sed` function in unix (Ritchie and Thompson 1978) and allows generic text pattern matching and replacing.
- `patch_code_env()`: replaces the common code environments, `code`, `example`, `Sin`, `Sout`, `Scode`, `Sinput`, `smallverbatim`, `boxedverbatim`, `smallexample` with `verbatim`.
- `patch_equations()`: coordinates various equation environments.
- `patch_figure_env()`: coordinates various figure environments.
- `patch_table_env()`: coordinates table environments.

These functions are verbose and describe all the changes being made. They also create a backup of the original file before making the changes.

Post-processing using Lua filters

Lua (Ierusalimschy, Figueiredo, and Filho 1996) is a programming language, that is light-weight, fast, ideal for procedural operations. It is embedded in many other applications to allow custom scripting for extensibility. Pandoc allows users to provide custom lua filters to produce custom output formats.

The texor package handles post-processing of the R Markdown document into the special format for the R Journal using a suite of Lua filters.

Here is an example of a Lua filter available in texor:

```
function Div(el)
    if el.classes[1] == 'thebibliography' then
        return { }
    end
end
```

This filter reads the abstract syntax tree, selecting all the Div elements. Then it looks for the class "thebibliography." This Div element contains the LaTeX bibliographic records, that appear at the very end of papers. It should not be in the document when using the "RJ-web-article" layout, because it is added from meta-data when the R Markdown is knitted. So the Lua filter removes this section.

Handling figures

Handling equations

Handling the bibliography

Logging the conversion

One-step user-facing conversion

The only function that a user necessarily needs is `latex_to_web()`. This creates the R Journal style R Markdown file from a given R Journal style LaTeX file. This is achieved in several conversion steps, `convert_to_markdown()`, `generate_rmd()` and `produce_html()`.

3 Using texor

The package `texor` can be installed from CRAN using:

```
install.packages("texor")
```

and the development version from https://abhi-1u.github.io/texor.

Examples / getting started / usage and so on

Run through the different examples in supplementary here

From general LaTeX to generic R Markdown

4 Managing the bibliography using rebib

Overview

The rebib package addresses the issue with LaTeX articles using built-in bibliography options with or without BibTeX files. While this works well with LaTeX, it won't work with Rmarkdown. Initially the goal was to use external software like Biber to convert the embedded bibliography to BibTeX. However the integration of external software was not viable, hence the experimental idea of a bibliography parser gained momentum.

It was initially a part of the texor package, as those functions grew in features and became more involved; at that point, it made sense to move those functions as a separate package. Initially, there were some reservations about the usage of rebib and its stability with various formats. However, the package has improved over time and proven to be a good performer.

Using rebib

Install instructions

Examples / getting started / usage and so on

5 Summary

Where this might be useful in the future, other applications.

6 Acknowledgments

pandoc "TODO : Later"

7 Supplementary materials

Several supplementary example documents are provided which illustrate how different elements of LaTeX-authored papers are handled. These are:

-

The GitHub repos supporting this work are:

- **texor**: https://abhi-1u.github.io/texor
- **rebib**: https://abhi-1u.github.io/rebib/
- This paper: https://github.com/Abhi-1U/texor-rjarticle
- More details on **rjtools** are at https://rjournal.github.io/rjtools/

References

Dervieux, Christophe, JJ Allaire, Rich Iannone, Alison Presmanes Hill, and Yihui Xie. 2022. *Distill: 'R Markdown' Format for Scientific and Technical Writing*. https://CRAN.R-project.org/package=distill.

Gruber, John. 2002. "Markdown." https://daringfireball.net/projects/markdown/.

Ierusalimschy, Roberto, Luiz Henrique de Figueiredo, and Waldemar Celes Filho. 1996. "Lua—an Extensible Extension Language." *Software: Practice and Experience* 26 (6): 635–52.

Josh Cassidy. 2013. *LaTeX Graphics using TikZ: A Tutorial for Beginners (Part 3)—Creating Flowcharts*. Overleaf tutorials. https://www.overleaf.com/learn/latex/.

MacFarlane, John. 2023. "Pandoc: A Universal Document Converter." https://pandoc.org.

Ritchie, O. M., and K. Thompson. 1978. "The UNIX Time-Sharing System." *The Bell System Technical Journal* 57 (6): 1905–29. https://doi.org/10.1002/j.1538-7305.1978.tb02136.x.

The LaTeX Project. 2023. "LaTeX – a Document Preparation System." https://www.latex-project.org.

Xie, Yihui. 2015. *Dynamic Documents with R and Knitr*. 2nd ed. Boca Raton, Florida: Chapman; Hall/CRC. https://yihui.org/knitr/.

Xie, Yihui, J. J. Allaire, and Garrett Grolemund. 2018. *R Markdown: The Definitive Guide*. Boca Raton, Florida: Chapman; Hall/CRC. https://bookdown.org/yihui/rmarkdown.

Abhishek Ulayil
Institute of Actuaries of India (student)
Mumbai, India
ORCiD: 0009-0000-6935-8690
perricoq@outlook.com

Mitch O'Hara-Wild
Monash University
Melbourne, Australia
ORCiD: 0000-0001-6729-7695
mail@mitchelloharawild.com

Christophe Dervieux
Posit PBC
Paris, France
ORCiD: 0000-0003-4474-2498
christophe.dervieux@gmail.com

Heather Turner
University of Warwick

Newport, United Kingdom
ORCiD: 0000-0002-1256-3375
ht@heatherturner.net

Dianne Cook
Monash University
Melbourne, Australia
ORCiD: 0000-0002-3813-7155
dicook@monash.edu