Writing R Markdown documents in Stata

Roger B. Newson
r.newson@qmul.ac.uk
http://www.rogernewsonresources.org.uk

Cancer Prevention Group, Wolfson Institute of Population Health, Queen Mary University of London

Presented at the 2026 UK Stata Conference, London,
03–04 September, 2026
Downloadable from the conference website at
https://econpapers.repec.org/paper/boc/lsug26/

What is R Markdown?

- A Markdown language is a shorthand language for HTML, and possibly for other languages.
- R Markdown[1] is a particularly good Markdown, implemented using the `knitr` function in R and the Pandoc application.
- `knitr` inputs a text document written in the R Markdown language, and can output documents in the formats HTML, `.docx`, and `.pdf` (via LaTeX).
- And it can make documents containing not only tables and graphics in all 3 formats, but also equations, using MathJax for HTML documents and LaTeX for `.pdf` documents.
- And, unsurprisingly, R Markdown documents can be written in Stata.
- Note that Pandoc is bundled into RStudio, and not everybody is allowed to download Pandoc without the rest of RStudio!

What is R Markdown?

- A Markdown language is a shorthand language for HTML, and possibly for other languages.
- R Markdown[1] is a particularly good Markdown, implemented using the `knitr` function in R and the Pandoc application.
- `knitr` inputs a text document written in the R Markdown language, and can output documents in the formats HTML, `.docx`, and `.pdf` (via LaTeX).
- And it can make documents containing not only tables and graphics in all 3 formats, but also equations, using MathJax for HTML documents and LaTeX for `.pdf` documents.
- And, unsurprisingly, R Markdown documents can be written in Stata.
- Note that Pandoc is bundled into RStudio, and not everybody is allowed to download Pandoc without the rest of RStudio!

What is R Markdown?

- A Markdown language is a shorthand language for HTML, and possibly for other languages.
- R Markdown[1] is a particularly good Markdown, implemented using the `knitr` function in R and the Pandoc application.
- `knitr` inputs a text document written in the R Markdown language, and can output documents in the formats HTML, `.docx`, and `.pdf` (via LaTeX).
- And it can make documents containing not only tables and graphics in all 3 formats, but also equations, using MathJax for HTML documents and LaTeX for `.pdf` documents.
- And, unsurprisingly, R Markdown documents can be written in Stata.
- Note that Pandoc is bundled into RStudio, and not everybody is allowed to download Pandoc without the rest of RStudio!

What is R Markdown?

- A Markdown language is a shorthand language for HTML, and possibly for other languages.
- R Markdown[1] is a particularly good Markdown, implemented using the `knitr` function in R and the Pandoc application.
- `knitr` inputs a text document written in the R Markdown language, and can output documents in the formats HTML, `.docx`, and `.pdf` (via LaTeX).
- And it can make documents containing not only tables and graphics in all 3 formats, but also equations, using MathJax for HTML documents and LaTeX for `.pdf` documents.
- And, unsurprisingly, R Markdown documents can be written in Stata.
- Note that Pandoc is bundled into RStudio, and not everybody is allowed to download Pandoc without the rest of RStudio!

What is R Markdown?

- A Markdown language is a shorthand language for HTML, and possibly for other languages.
- R Markdown[1] is a particularly good Markdown, implemented using the `knitr` function in R and the Pandoc application.
- `knitr` inputs a text document written in the R Markdown language, and can output documents in the formats HTML, `.docx`, and `.pdf` (via LaTeX).
- And it can make documents containing not only tables and graphics in all 3 formats, but also equations, using MathJax for HTML documents and LaTeX for `.pdf` documents.
- And, unsurprisingly, R Markdown documents can be written in Stata.
- Note that Pandoc is bundled into RStudio, and not everybody is allowed to download Pandoc without the rest of RStudio!

What is R Markdown?

- A Markdown language is a shorthand language for HTML, and possibly for other languages.
- R Markdown[1] is a particularly good Markdown, implemented using the `knitr` function in R and the Pandoc application.
- `knitr` inputs a text document written in the R Markdown language, and can output documents in the formats HTML, `.docx`, and `.pdf` (via LaTeX).
- And it can make documents containing not only tables and graphics in all 3 formats, but also equations, using MathJax for HTML documents and LaTeX for `.pdf` documents.
- And, unsurprisingly, R Markdown documents can be written in Stata.
- Note that Pandoc is bundled into RStudio, and not everybody is allowed to download Pandoc without the rest of RStudio!

What is R Markdown?

- A Markdown language is a shorthand language for HTML, and possibly for other languages.
- R Markdown[1] is a particularly good Markdown, implemented using the `knitr` function in R and the Pandoc application.
- `knitr` inputs a text document written in the R Markdown language, and can output documents in the formats HTML, `.docx`, and `.pdf` (via LaTeX).
- And it can make documents containing not only tables and graphics in all 3 formats, but also equations, using MathJax for HTML documents and LaTeX for `.pdf` documents.
- And, unsurprisingly, R Markdown documents can be written in Stata.
- Note that Pandoc is bundled into RStudio, and not everybody is allowed to download Pandoc without the rest of RStudio!

So how do we write R Markdown documents in Stata?

- We use the `file` utility of official Stata.
- To make the tables, we use the SSC package `listtab`[2], with the row style `rstyle(markdown)` [3].
- To make the rest of the document, we use the SSC package `tfinsert` to insert streams of text from a file or from the command input (using a `terminator(`*`string`*`)` option).
- We will demonstrate this using an example in the dataset created by the SSC package `xauto`.

So how do we write R Markdown documents in Stata?

- We use the `file` utility of official Stata.
- To make the tables, we use the SSC package `listtab`[2], with the row style `rstyle(markdown)` [3].
- To make the rest of the document, we use the SSC package `tfinsert` to insert streams of text from a file or from the command input (using a `terminator(`*`string`*`)` option).
- We will demonstrate this using an example in the dataset created by the SSC package `xauto`.

So how do we write R Markdown documents in Stata?

- We use the `file` utility of official Stata.
- To make the tables, we use the SSC package `listtab`[2], with the row style `rstyle(markdown)` [3].
- To make the rest of the document, we use the SSC package `tfinsert` to insert streams of text from a file or from the command input (using a `terminator(`*`string`*`)` option).
- We will demonstrate this using an example in the dataset created by the SSC package `xauto`.

So how do we write R Markdown documents in Stata?

- We use the `file` utility of official Stata.
- To make the tables, we use the SSC package `listtab`[2], with the row style `rstyle(markdown)` [3].
- To make the rest of the document, we use the SSC package `tfinsert` to insert streams of text from a file or from the command input (using a `terminator(`*`string`*`)` option).
- We will demonstrate this using an example in the dataset created by the SSC package `xauto`.

So how do we write R Markdown documents in Stata?

- We use the `file` utility of official Stata.
- To make the tables, we use the SSC package `listtab`[2], with the row style `rstyle(markdown)` [3].
- To make the rest of the document, we use the SSC package `tfinsert` to insert streams of text from a file or from the command input (using a `terminator(`*`string`*`)` option).
- We will demonstrate this using an example in the dataset created by the SSC package `xauto`.

`example1.do`: An example in the `xauto` data

- The `xauto` data is an extended version of the `auto` data, with extra variables.
- We will use the ***Mata–based*** package `rcentile`[4] to estimate percentiles of `tons` (car weight in US tons), clustered by the variable `firm` (the firm that makes the car model).
- So we assume that we are sampling firms from a population of firms, instead of models from a population of models.
- We will use the SSC package `xsvmat` to save the confidence intervals for the percentiles to a resultsset.
- We will then write the stats to a `.Rmd` (R Markdown) document, containing equations, a Figure, and a Table.
- The `.Rmd` document will then be converted to a HTML document.

`example1.do`: An example in the `xauto` data

- The `xauto` data is an extended version of the `auto` data, with extra variables.
- We will use the ***Mata–based*** package `rcentile`[4] to estimate percentiles of `tons` (car weight in US tons), clustered by the variable `firm` (the firm that makes the car model).
- So we assume that we are sampling firms from a population of firms, instead of models from a population of models.
- We will use the SSC package `xsvmat` to save the confidence intervals for the percentiles to a resultsset.
- We will then write the stats to a `.Rmd` (R Markdown) document, containing equations, a Figure, and a Table.
- The `.Rmd` document will then be converted to a HTML document.

`example1.do`: An example in the `xauto` data

- The `xauto` data is an extended version of the `auto` data, with extra variables.
- We will use the ***Mata–based*** package `rcentile`[4] to estimate percentiles of `tons` (car weight in US tons), clustered by the variable `firm` (the firm that makes the car model).
- So we assume that we are sampling firms from a population of firms, instead of models from a population of models.
- We will use the SSC package `xsvmat` to save the confidence intervals for the percentiles to a resultsset.
- We will then write the stats to a `.Rmd` (R Markdown) document, containing equations, a Figure, and a Table.
- The `.Rmd` document will then be converted to a HTML document.

`example1.do`: An example in the `xauto` data

- The `xauto` data is an extended version of the `auto` data, with extra variables.
- We will use the ***Mata–based*** package `rcentile`[4] to estimate percentiles of `tons` (car weight in US tons), clustered by the variable `firm` (the firm that makes the car model).
- So we assume that we are sampling firms from a population of firms, instead of models from a population of models.
- We will use the SSC package `xsvmat` to save the confidence intervals for the percentiles to a resultsset.
- We will then write the stats to a `.Rmd` (R Markdown) document, containing equations, a Figure, and a Table.
- The `.Rmd` document will then be converted to a HTML document.

`example1.do`: An example in the `xauto` data

- The `xauto` data is an extended version of the `auto` data, with extra variables.
- We will use the ***Mata–based*** package `rcentile`[4] to estimate percentiles of `tons` (car weight in US tons), clustered by the variable `firm` (the firm that makes the car model).
- So we assume that we are sampling firms from a population of firms, instead of models from a population of models.
- We will use the SSC package `xsvmat` to save the confidence intervals for the percentiles to a resultsset.
- We will then write the stats to a `.Rmd` (R Markdown) document, containing equations, a Figure, and a Table.
- The `.Rmd` document will then be converted to a HTML document.

`example1.do`: An example in the `xauto` data

- The `xauto` data is an extended version of the `auto` data, with extra variables.
- We will use the ***Mata–based*** package `rcentile`[4] to estimate percentiles of `tons` (car weight in US tons), clustered by the variable `firm` (the firm that makes the car model).
- So we assume that we are sampling firms from a population of firms, instead of models from a population of models.
- We will use the SSC package `xsvmat` to save the confidence intervals for the percentiles to a resultsset.
- We will then write the stats to a `.Rmd` (R Markdown) document, containing equations, a Figure, and a Table.
- The `.Rmd` document will then be converted to a HTML document.

`example1.do`: An example in the `xauto` data

- The `xauto` data is an extended version of the `auto` data, with extra variables.
- We will use the ***Mata–based*** package `rcentile`[4] to estimate percentiles of `tons` (car weight in US tons), clustered by the variable `firm` (the firm that makes the car model).
- So we assume that we are sampling firms from a population of firms, instead of models from a population of models.
- We will use the SSC package `xsvmat` to save the confidence intervals for the percentiles to a resultsset.
- We will then write the stats to a `.Rmd` (R Markdown) document, containing equations, a Figure, and a Table.
- The `.Rmd` document will then be converted to a HTML document.

Estimating the percentiles of `tons`

In the `xauto` data, we compute confidence intervals for percentiles 0 by 25 to 100 for the variable `tons`, by inverting confidence intervals for the Fisher z–transform of the mean sign difference $E[\text{sign}(Y-\theta)]$, where $E[\cdot]$ denotes expectation, Y is `tons`, and θ is the percentile:

```
. rcentile tons, cluster(firm) centile(0(25)100);
Percentile(s) for variable: tons
Mean sign transformation: Fisher's z
Valid observations: 74
Number of clusters (firm) = 23
95% confidence interval(s) for percentile(s)
  Percent    Centile    Minimum    Maximum
        0        .88        .88        .88
       25       1.12      1.035      1.365
       50      1.595      1.325      1.685
       75        1.8      1.655       1.94
      100       2.42       2.42       2.42
```

We see that we have sampled 23 firms, with a total of 74 car models.

The next step: Save results and a resultsset

We then save the results `r(N_clust)` and `r(N)`, containing numbers of firms and models respectively, in local macros `Nfirm` and `Nmodel` respectively, and use the SSC package `xsvmat` to save the CI matrix result `r(cimat)` in a resultsset in memory, overwriting the original dataset, and list the resultsset:

```
. local Nfirm=r(N_clust);

. local Nmodel=r(N);

. xsvmat, from(r(cimat)) fast names(col) format(Centile Minimum Maximum %8.3f)
>   list(, abbr(32));

     +---------------------------------------+
     | Percent   Centile   Minimum   Maximum |
     |---------------------------------------|
  1. |       0     0.880     0.880     0.880 |
  2. |      25     1.120     1.035     1.365 |
  3. |      50     1.595     1.325     1.685 |
  4. |      75     1.800     1.655     1.940 |
  5. |     100     2.420     2.420     2.420 |
     +---------------------------------------+
```

We see that the percents, percentiles, and confidence limits are now neatly formatted!

So now we can make the R Markdown document

- We will do this with a `file open` command, followed by a `capture noisily` block[5], followed by a `file close` command.
- Inside the `capture noisily` block, we will generate tables using the `listtab` command with the option `rstyle(markdown)`, and the rest of the document using `file write` and the SSC package `tfinsert`.
- R Markdown documents (unlike Stata Markdown documents) start with an introduction section, written in a language called YAML ("YAML Ain't Markup Language").
- This YAML section specifies the title, the author, the document format (HTML, PDF, or `.docx`), and sometimes other things.

So now we can make the R Markdown document

- We will do this with a `file open` command, followed by a `capture noisily` block[5], followed by a `file close` command.
- Inside the `capture noisily` block, we will generate tables using the `listtab` command with the option `rstyle(markdown)`, and the rest of the document using `file write` and the SSC package `tfinsert`.
- R Markdown documents (unlike Stata Markdown documents) start with an introduction section, written in a language called YAML ("YAML Ain't Markup Language").
- This YAML section specifies the title, the author, the document format (HTML, PDF, or `.docx`), and sometimes other things.

So now we can make the R Markdown document

- We will do this with a `file open` command, followed by a `capture noisily` block[5], followed by a `file close` command.
- Inside the `capture noisily` block, we will generate tables using the `listtab` command with the option `rstyle(markdown)`, and the rest of the document using `file write` and the SSC package `tfinsert`.
- R Markdown documents (unlike Stata Markdown documents) start with an introduction section, written in a language called YAML ("YAML Ain't Markup Language").
- This YAML section specifies the title, the author, the document format (HTML, PDF, or `.docx`), and sometimes other things.

So now we can make the R Markdown document

- We will do this with a `file open` command, followed by a `capture noisily` block[5], followed by a `file close` command.
- Inside the `capture noisily` block, we will generate tables using the `listtab` command with the option `rstyle(markdown)`, and the rest of the document using `file write` and the SSC package `tfinsert`.
- R Markdown documents (unlike Stata Markdown documents) start with an introduction section, written in a language called YAML ("YAML Ain't Markup Language").
- This YAML section specifies the title, the author, the document format (HTML, PDF, or `.docx`), and sometimes other things.

So now we can make the R Markdown document

- We will do this with a `file open` command, followed by a `capture noisily` block[5], followed by a `file close` command.
- Inside the `capture noisily` block, we will generate tables using the `listtab` command with the option `rstyle(markdown)`, and the rest of the document using `file write` and the SSC package `tfinsert`.
- R Markdown documents (unlike Stata Markdown documents) start with an introduction section, written in a language called YAML ("YAML Ain't Markup Language").
- This YAML section specifies the title, the author, the document format (HTML, PDF, or `.docx`), and sometimes other things.

Making a document using a `capture noisily` block

We start the document with a `file open` command, and then start a `capture noisily` block:

```
* Beginning of document creation section *;
tempname rmdhandle;
file open `rmdhandle' using example1.Rmd,
  write text replace;
capture noisily {;
```

And we end the document when we end the `capture noisily` block, and then close the document with a `file close` command:

```
};
file close `rmdhandle';
```

This practice ensures that, if a command fails inside the `capture noisily` block, then the file being created will be created, closed, and available for our inspection.

Inside the `capture noisily` block: Writing the YAML metadata

The SSC package `tfinsert` inputs an instream sequence of lines, terminated as specified by the option `terminator(END_OF_YAML)`. The instream metadataset starts and ends with `---` and specifies the title, author, and document type.

```
* Add YAML metadata *;
#delim cr
tfinsert `rmdhandle', terminator(END_OF_YAML) maxlines(999999)
---
title:  Percentiles of weight in tons in the `xauto` data
author: Roger B. Newson
output: html_document
---
END_OF_YAML
#delim ;
```

Note that `tfinsert` can only read an instream sequence of lines when `#delim cr` is specified.

Inside the `capture noisily` block: Writing some text

We then insert some R Markdown text, again using `tfinsert`:

```
#delim cr
tfinsert `rmdhandle', terminator(END_OF_RMD) maxlines(999999)

\newpage

Methods
-------

We used the Mata-based SSC package `rcentile` to estimate percentiles
0 to 100 by 25, with confidence intervals clustered by the variable `firm`,
meaning that we are sampling firms from a population of firms,
instead of models from a population of models. A 100$\alpha$th percentile
of a variable $Y$ is defined informally as a solution in $\theta$ of the equation
$$
C_Y(\theta)=\alpha,
$$
where $C_Y(\cdot)$ is the centre ridit function of $Y$.

Results
-------

The percentiles are plotted in Figure 1 and tabulated in Table 1.

END_OF_RMD

#delim ;
```

This is alien–looking and long–winded, but less so than HTML, LaTeX, or `putdocx`!

Inside the `capture noisily` block: Adding a Figure

We then add a Figure, using the SSC packages `regaxis` and `eclplot` to make a confidence interval plot of percentiles against percents, together with 2 `file writes` and a `graph export` in official Stata:

```
* Add Figure *;
file write `rmdhandle' _n "\newpage"
  _n _n
  "## Figure 1. Percentiles of weight in tons in the `xauto` data "
  "(`Nfirm' firms, `Nmodel' models)"
  _n _n;
regaxis Centile Minimum Maximum, ltick(xlabs) inc(0);
eclplot Centile Minimum Maximum Percent, hori eplot(connected) rplot(rcap)
  estopts(sort)
  ylab(0(25)100)
  xlab(`xlabs', format(%-8.1f))
  xtitle("Percentile (tons) with 95% CI")
  xsize(8) ysize(6);
graph export figure1.svg, replace;
file write `rmdhandle' _n "![](figure1.svg)";
```

Note that the Figure is a HTML–friendly `.svg` file.

Inside the `capture noisily` block: Adding a Table

We then add a Table, using `file write`, `preserve`, and `restore` in official Stata, together with the SSC packages `bmjcip`, `sdecode`, `chardef`, and `listtab` (between the `preserve` and the `restore`), to make a confidence interval table of percents and percentiles:

```
. file write `rmdhandle' _n "\newpage"
>   _n _n
>   "## Table 1. Percentiles of weight in tons in the `xauto` data "
>   "(`Nfirm' firms, `Nmodel' models)"
>   _n _n;
. preserve;
. bmjcip Centile Minimum Maximum;
. sdecode Percent, replace;
. chardef Percent Centile Minimum Maximum,
>  char(varname) val(Percent Percentile "(95%" "CI)") pref(*) suff(*);
. chardef Percent Centile Minimum Maximum,
>   char(halign) val(---:);
. listtab Percent Centile Minimum Maximum, handle("`rmdhandle'")
>   rstyle(markdown) headc(varname halign);
. restore;
```

This code is a simple example of the "decode and list" methods of Newson (2023)[3]. and Newson (2012)[2].

After the `capture noisily` block: Converting the R Markdown document to HTML

- The `capture noisily` block, together with the `file open` command before it and the `file close` command after it, have created a R Markdown (`.Rmd`) document `example1.Rmd`.
- We can convert this to a HTML document `example1.html`, using RStudio, which contains Pandoc and the `knitr` function.
- And then we can then view our HTML document using a browser.

After the `capture noisily` block: Converting the R Markdown document to HTML

- The `capture noisily` block, together with the `file open` command before it and the `file close` command after it, have created a R Markdown (`.Rmd`) document `example1.Rmd`.
- We can convert this to a HTML document `example1.html`, using RStudio, which contains Pandoc and the `knitr` function.
- And then we can then view our HTML document using a browser.

After the `capture noisily` block: Converting the R Markdown document to HTML

- The `capture noisily` block, together with the `file open` command before it and the `file close` command after it, have created a R Markdown (`.Rmd`) document `example1.Rmd`.
- We can convert this to a HTML document `example1.html`, using RStudio, which contains Pandoc and the `knitr` function.
- And then we can then view our HTML document using a browser.

After the `capture noisily` block: Converting the R Markdown document to HTML

- The `capture noisily` block, together with the `file open` command before it and the `file close` command after it, have created a R Markdown (`.Rmd`) document `example1.Rmd`.
- We can convert this to a HTML document `example1.html`, using RStudio, which contains Pandoc and the `knitr` function.
- And then we can then view our HTML document using a browser.

`example2.do`: Making a `.pdf` document via LaTeX

- Instead of a HTML document, R Markdown can make a `.pdf` document, using LaTeX.
- This time, the do–file is `example2.do`, and the R Markdown document is `example2.Rmd`.
- And RStudio creates a temporary LaTeX document, which is converted into a PDF file `example2.pdf`.
- And, this time, the figure lives in a file `figure1.pdf`.
- We can now have a look at `example2.pdf`.

`example2.do`: Making a `.pdf` document via LaTeX

- Instead of a HTML document, R Markdown can make a `.pdf` document, using LaTeX.
- This time, the do–file is `example2.do`, and the R Markdown document is `example2.Rmd`.
- And RStudio creates a temporary LaTeX document, which is converted into a PDF file `example2.pdf`.
- And, this time, the figure lives in a file `figure1.pdf`.
- We can now have a look at `example2.pdf`.

`example2.do`: Making a `.pdf` document via LaTeX

- Instead of a HTML document, R Markdown can make a `.pdf` document, using LaTeX.
- This time, the do–file is `example2.do`, and the R Markdown document is `example2.Rmd`.
- And RStudio creates a temporary LaTeX document, which is converted into a PDF file `example2.pdf`.
- And, this time, the figure lives in a file `figure1.pdf`.
- We can now have a look at `example2.pdf`.

`example2.do`: Making a `.pdf` document via LaTeX

- Instead of a HTML document, R Markdown can make a `.pdf` document, using LaTeX.
- This time, the do–file is `example2.do`, and the R Markdown document is `example2.Rmd`.
- And RStudio creates a temporary LaTeX document, which is converted into a PDF file `example2.pdf`.
- And, this time, the figure lives in a file `figure1.pdf`.
- We can now have a look at `example2.pdf`.

`example2.do`: Making a `.pdf` document via LaTeX

- Instead of a HTML document, R Markdown can make a `.pdf` document, using LaTeX.
- This time, the do–file is `example2.do`, and the R Markdown document is `example2.Rmd`.
- And RStudio creates a temporary LaTeX document, which is converted into a PDF file `example2.pdf`.
- And, this time, the figure lives in a file `figure1.pdf`.
- We can now have a look at `example2.pdf`.

`example2.do`: Making a `.pdf` document via LaTeX

- Instead of a HTML document, R Markdown can make a `.pdf` document, using LaTeX.
- This time, the do–file is `example2.do`, and the R Markdown document is `example2.Rmd`.
- And RStudio creates a temporary LaTeX document, which is converted into a PDF file `example2.pdf`.
- And, this time, the figure lives in a file `figure1.pdf`.
- We can now have a look at `example2.pdf`.

`example3.do`: Making a `.docx` document

- R Markdown can also make a `.docx` document, openable with Microsoft Word.
- This time, the do–file is `example3.do`, and the R Markdown document is `example3.Rmd`, which is converted into a `.docx` file `example3.docx`.
- In this case, we specify the style of the `.docx` document using a **template document**, which is named in the YAML preamble.
- The template document is created by starting with an empty Pandoc–generated `.docx` file, and manually editing the various styles, including the header and the footer.
- In our case, we will call the template document `example3_template.docx`, and have a look at it.

`example3.do`: Making a `.docx` document

- R Markdown can also make a `.docx` document, openable with Microsoft Word.
- This time, the do–file is `example3.do`, and the R Markdown document is `example3.Rmd`, which is converted into a `.docx` file `example3.docx`.
- In this case, we specify the style of the `.docx` document using a **template document**, which is named in the YAML preamble.
- The template document is created by starting with an empty Pandoc–generated `.docx` file, and manually editing the various styles, including the header and the footer.
- In our case, we will call the template document `example3_template.docx`, and have a look at it.

`example3.do`: Making a `.docx` document

- R Markdown can also make a `.docx` document, openable with Microsoft Word.
- This time, the do–file is `example3.do`, and the R Markdown document is `example3.Rmd`, which is converted into a `.docx` file `example3.docx`.
- In this case, we specify the style of the `.docx` document using a **template document**, which is named in the YAML preamble.
- The template document is created by starting with an empty Pandoc–generated `.docx` file, and manually editing the various styles, including the header and the footer.
- In our case, we will call the template document `example3_template.docx`, and have a look at it.

`example3.do`: Making a `.docx` document

- R Markdown can also make a `.docx` document, openable with Microsoft Word.
- This time, the do–file is `example3.do`, and the R Markdown document is `example3.Rmd`, which is converted into a `.docx` file `example3.docx`.
- In this case, we specify the style of the `.docx` document using a **template document**, which is named in the YAML preamble.
- The template document is created by starting with an empty Pandoc–generated `.docx` file, and manually editing the various styles, including the header and the footer.
- In our case, we will call the template document `example3_template.docx`, and have a look at it.

`example3.do`: Making a `.docx` document

- R Markdown can also make a `.docx` document, openable with Microsoft Word.
- This time, the do–file is `example3.do`, and the R Markdown document is `example3.Rmd`, which is converted into a `.docx` file `example3.docx`.
- In this case, we specify the style of the `.docx` document using a **template document**, which is named in the YAML preamble.
- The template document is created by starting with an empty Pandoc–generated `.docx` file, and manually editing the various styles, including the header and the footer.
- In our case, we will call the template document `example3_template.docx`, and have a look at it.

`example3.do`: Making a `.docx` document

- R Markdown can also make a `.docx` document, openable with Microsoft Word.
- This time, the do–file is `example3.do`, and the R Markdown document is `example3.Rmd`, which is converted into a `.docx` file `example3.docx`.
- In this case, we specify the style of the `.docx` document using a **template document**, which is named in the YAML preamble.
- The template document is created by starting with an empty Pandoc–generated `.docx` file, and manually editing the various styles, including the header and the footer.
- In our case, we will call the template document `example3_template.docx`, and have a look at it.

The YAML metadata for `example3.Rmd`

When the percentiles of the variable `tons` have been estimated, we start writing the R Markdown file `example3.Rmd`, starting with a slightly more complicated preamble of YAML metadata:

```
* Add YAML metadata *;
#delim cr
tfinsert `rmdhandle', terminator(END_OF_YAML) maxlines(999999)
---
title:  Percentiles of weight in tons in the `xauto` data
author: Roger B. Newson
output:
  word_document:
    reference_docx: "example3_template.docx"
---
END_OF_YAML
#delim ;
```

This time, we plan to output to a Word document `example3.docx`, created using the template `example3_template.docx` that we made earlier. Note that indentation is significant in YAML!

Converting the R Markdown document to `.docx`

- The do–file `example3.do` creates a R Markdown file `example3.Rmd`.
- This time, Figure 1 is created as an Enhanced Metafile `figure1.emf`.
- We can then convert `example3.Rmd` to a `.docx` document `example3.docx`, using RStudio.
- And then we can then view `example3.docx`.
- Note that the style elements (including the table style). and the header and the footer, have been imported from the template file `example3_template.docx`.
- Note also that the `.docx` version contains the math formulas, as the `.html` and the `.pdf` versions did!

Converting the R Markdown document to `.docx`

- The do–file `example3.do` creates a R Markdown file `example3.Rmd`.
- This time, Figure 1 is created as an Enhanced Metafile `figure1.emf`.
- We can then convert `example3.Rmd` to a `.docx` document `example3.docx`, using RStudio.
- And then we can then view `example3.docx`.
- Note that the style elements (including the table style). and the header and the footer, have been imported from the template file `example3_template.docx`.
- Note also that the `.docx` version contains the math formulas, as the `.html` and the `.pdf` versions did!

Converting the R Markdown document to `.docx`

- The do–file `example3.do` creates a R Markdown file `example3.Rmd`.
- This time, Figure 1 is created as an Enhanced Metafile `figure1.emf`.
- We can then convert `example3.Rmd` to a `.docx` document `example3.docx`, using RStudio.
- And then we can then view `example3.docx`.
- Note that the style elements (including the table style). and the header and the footer, have been imported from the template file `example3_template.docx`.
- Note also that the `.docx` version contains the math formulas, as the `.html` and the `.pdf` versions did!

Converting the R Markdown document to `.docx`

- The do–file `example3.do` creates a R Markdown file `example3.Rmd`.
- This time, Figure 1 is created as an Enhanced Metafile `figure1.emf`.
- We can then convert `example3.Rmd` to a `.docx` document `example3.docx`, using RStudio.
- And then we can then view `example3.docx`.
- Note that the style elements (including the table style). and the header and the footer, have been imported from the template file `example3_template.docx`.
- Note also that the `.docx` version contains the math formulas, as the `.html` and the `.pdf` versions did!

Converting the R Markdown document to `.docx`

- The do–file `example3.do` creates a R Markdown file `example3.Rmd`.
- This time, Figure 1 is created as an Enhanced Metafile `figure1.emf`.
- We can then convert `example3.Rmd` to a `.docx` document `example3.docx`, using RStudio.
- And then we can then view `example3.docx`.
- Note that the style elements (including the table style). and the header and the footer, have been imported from the template file `example3_template.docx`.
- Note also that the `.docx` version contains the math formulas, as the `.html` and the `.pdf` versions did!

Converting the R Markdown document to `.docx`

- The do–file `example3.do` creates a R Markdown file `example3.Rmd`.
- This time, Figure 1 is created as an Enhanced Metafile `figure1.emf`.
- We can then convert `example3.Rmd` to a `.docx` document `example3.docx`, using RStudio.
- And then we can then view `example3.docx`.
- Note that the style elements (including the table style). and the header and the footer, have been imported from the template file `example3_template.docx`.
- Note also that the `.docx` version contains the math formulas, as the `.html` and the `.pdf` versions did!

Converting the R Markdown document to `.docx`

- The do–file `example3.do` creates a R Markdown file `example3.Rmd`.
- This time, Figure 1 is created as an Enhanced Metafile `figure1.emf`.
- We can then convert `example3.Rmd` to a `.docx` document `example3.docx`, using RStudio.
- And then we can then view `example3.docx`.
- Note that the style elements (including the table style). and the header and the footer, have been imported from the template file `example3_template.docx`.
- Note also that the `.docx` version contains the math formulas, as the `.html` and the `.pdf` versions did!

Further possibilities

- R Markdown (and more generally Pandoc) supplies a *de luxe* Markdown, inputting text files, and giving the possibility of equations, and the choice of PDF, HTML, and `.docx` outputs.
- And the input text document can easily be created using Stata, using `file` with `tfinsert` and `listtab`.
- More complicated long–hand HTML and LaTeX tables could be made, using `listtab` with the options `rstyle(html)` or `rstyle(tabular)`, respectively.
- It *might* be a good idea to bundle Pandoc into Stata, if that would be legal. (That way, Pandoc would be available to *all* Stata users, by–passing RStudio!)
- All the above is work in progress. I have no plans to abandon `putdocx` and the SSC package `docxtab`[3], which make neater `.docx` tables!

Further possibilities

- R Markdown (and more generally Pandoc) supplies a *de luxe* Markdown, inputting text files, and giving the possibility of equations, and the choice of PDF, HTML, and `.docx` outputs.
- And the input text document can easily be created using Stata, using `file` with `tfinsert` and `listtab`.
- More complicated long–hand HTML and LaTeX tables could be made, using `listtab` with the options `rstyle(html)` or `rstyle(tabular)`, respectively.
- It *might* be a good idea to bundle Pandoc into Stata, if that would be legal. (That way, Pandoc would be available to *all* Stata users, by–passing RStudio!)
- All the above is work in progress. I have no plans to abandon `putdocx` and the SSC package `docxtab`[3], which make neater `.docx` tables!

Further possibilities

- R Markdown (and more generally Pandoc) supplies a *de luxe* Markdown, inputting text files, and giving the possibility of equations, and the choice of PDF, HTML, and `.docx` outputs.
- And the input text document can easily be created using Stata, using `file` with `tfinsert` and `listtab`.
- More complicated long–hand HTML and LaTeX tables could be made, using `listtab` with the options `rstyle(html)` or `rstyle(tabular)`, respectively.
- It *might* be a good idea to bundle Pandoc into Stata, if that would be legal. (That way, Pandoc would be available to *all* Stata users, by–passing RStudio!)
- All the above is work in progress. I have no plans to abandon `putdocx` and the SSC package `docxtab`[3], which make neater `.docx` tables!

Further possibilities

- R Markdown (and more generally Pandoc) supplies a *de luxe* Markdown, inputting text files, and giving the possibility of equations, and the choice of PDF, HTML, and `.docx` outputs.
- And the input text document can easily be created using Stata, using `file` with `tfinsert` and `listtab`.
- More complicated long–hand HTML and LaTeX tables could be made, using `listtab` with the options `rstyle(html)` or `rstyle(tabular)`, respectively.
- It *might* be a good idea to bundle Pandoc into Stata, if that would be legal. (That way, Pandoc would be available to *all* Stata users, by–passing RStudio!)
- All the above is work in progress. I have no plans to abandon `putdocx` and the SSC package `docxtab`[3], which make neater `.docx` tables!

Further possibilities

- R Markdown (and more generally Pandoc) supplies a *de luxe* Markdown, inputting text files, and giving the possibility of equations, and the choice of PDF, HTML, and `.docx` outputs.
- And the input text document can easily be created using Stata, using `file` with `tfinsert` and `listtab`.
- More complicated long–hand HTML and LaTeX tables could be made, using `listtab` with the options `rstyle(html)` or `rstyle(tabular)`, respectively.
- It *might* be a good idea to bundle Pandoc into Stata, if that would be legal. (That way, Pandoc would be available to *all* Stata users, by–passing RStudio!)
- All the above is work in progress. I have no plans to abandon `putdocx` and the SSC package `docxtab`[3], which make neater `.docx` tables!

Further possibilities

- R Markdown (and more generally Pandoc) supplies a *de luxe* Markdown, inputting text files, and giving the possibility of equations, and the choice of PDF, HTML, and `.docx` outputs.
- And the input text document can easily be created using Stata, using `file` with `tfinsert` and `listtab`.
- More complicated long–hand HTML and LaTeX tables could be made, using `listtab` with the options `rstyle(html)` or `rstyle(tabular)`, respectively.
- It *might* be a good idea to bundle Pandoc into Stata, if that would be legal. (That way, Pandoc would be available to *all* Stata users, by–passing RStudio!)
- All the above is work in progress. I have no plans to abandon `putdocx` and the SSC package `docxtab`[3], which make neater `.docx` tables!

References

[1] Xie, Y., Dervieux, C. and Riederer, E. 2021. *R Markdown Cookbook.* Boca Raton, CA: Chapman & Hall/CRC Press.

[2] Newson, R. B. 2012. From resultssets to resultstables in Stata. *The Stata Journal* 2012; **12(2)**: 191–213. Downloadable from *https://journals.sagepub.com/doi/pdf/10.1177/1536867X1201200203*

[3] Newson, R. B. Customized Markdown and `.docx` tables using `listtab` and `docxtab`. Presented at the 2023 London Stata Conference, 7–8 September, 2023. Downloadable from *https://econpapers.repec.org/paper/boclsug23/01.htm*

[4] Newson, R. B. Easy–to–use packages for estimating rank and spline parameters. Presented at the 20th UK Stata User Meeting, 11–12 September, 2014. Downloadable from *https://ideas.repec.org/p/boc/usug14/01.html*

[5] Newson, R. B. 2017. Stata tip 127: Use `capture noisily` groups. *The Stata Journal* 2017; **17(2)**: 511–514. Downloadable from *https://journals.sagepub.com/doi/pdf/10.1177/1536867X1701700215*

The presentation, and the example do–files, can be downloaded from the conference website. The packages can be downloaded from SSC.