Course title

Lecture title

Your name

University | Course code

Contents

Before you begin

This template is for knitting R Markdown documents to *both* HTML and PDF format. It tries to take care of various inconsistencies between the two formats with minimum effort from the user. Just click "Knit" (in Rstudio) and it will automatically export to both formats. As the name suggests, I predominantly use it for my lecture notes. But I find that it works well for writing papers too.

See the package README for a longer description, as well as potential gotchas and limitations (e.g. font support for different LaTeX engines).

Template features

Here are some examples of features not available in vanilla R Markdown and how to use them.

Multi-column environments

Multi-column environments are supported via's Pandoc's fenced_divs syntax and some preamble sugar (bundled together with the template). For example, a two-column section would look like this.

Here is some example **dplyr** code.

```
library(dplyr)

mtcars %>%
  group_by(am) %>%
  summarise(mean(mpg))
```

```
## # A tibble: 2 x 2
##      am `mean(mpg)`
##   <dbl>       <dbl>
## 1     0        17.1
## 2     1        24.4
```

And the **data.table** equivalent.

```
library(data.table)

mtcars_dt = as.data.table(mtcars)
mtcars_dt[, mean(mpg), by = am]
```

```
##        am       V1
##     <num>    <num>
## 1:      1 24.39231
## 2:      0 17.14737
```

The same idea can be extended to additional columns and the individual column widths are also adjustable.

Regression tables

I have fairly strong preferences about how regression tables should look (threeparttable FTW). Luckily, the fantastic **modelsummary** package has us covered for nice looking regression tables, particularly since it automatically supports different Rmd output formats and backends. (For example, via the equally excellent **kableExtra** package.) This makes it easy to produce regression tables that look good in both HTML and PDF… although the latter requires that the corresponding LaTeX packages be loaded first. This template loads those LaTeX packages automatically, so tables like the below Just Work^TM.

```r
library(modelsummary)
library(kableExtra)

mod = lm(mpg ~ vs * wt, mtcars)

msummary(
  mod,
  vcov = list("iid", "hc1", "hc2", "hc3", ~cyl), ## Multiple SEs for our model
  stars = TRUE,
  gof_omit = "^(?!Std)" ## Optional
  ) %>%
add_footnote(
    c("Wow, look at all those standard errors!",
      paste("This next footnote is pretty long. In fact, it runs over several",
            "lines of standard PDF output. Luckily that's no problem thanks to",
            "modelsummary, kableExtra, and threeparttable.")
      ),
    threeparttable = TRUE
    ) %>%
kable_styling(
    latex_options = "hold_position", ## (Optional) Print table directly below code
    position = "center"              ## (Optional) Center table
    )
```

	Model 1	Model 2	Model 3	Model 4	Model 5
(Intercept)	29.531***	29.531***	29.531***	29.531***	29.531***
	(2.622)	(2.710)	(2.877)	(3.270)	(4.022)
vs	11.767**	11.767**	11.767**	11.767*	11.767*
	(3.764)	(3.798)	(3.937)	(4.374)	(4.392)
wt	−3.501***	−3.501***	−3.501***	−3.501***	−3.501***
	(0.692)	(0.712)	(0.761)	(0.870)	(0.946)
vs × wt	−2.910*	−2.910*	−2.910*	−2.910*	−2.910*
	(1.216)	(1.113)	(1.153)	(1.280)	(1.167)
Std.Errors	IID	HC1	HC2	HC3	by: cyl

$+ p < 0.1, * p < 0.05, ** p < 0.01, *** p < 0.001$

[a] Wow, look at all those standard errors!

[b] This next footnote is pretty long. In fact, it runs over several lines of standard PDF output. Luckily that's no problem thanks to modelsummary, kableExtra, and threeparttable.

PDF support for non-standard fonts

This is an easy one; simply a matter of adding dev: cairo_pdf to the YAML. But it's nice not having to remember that every time, no?

Note: As the figure caption suggests, to run this next chunk you'll need to add Arial Narrow to your font book if it's not installed

on your system already.

```
library(ggplot2)
library(hrbrthemes)

ggplot(mtcars, aes(mpg, wt)) +
  geom_point() +
  labs(x = "Fuel efficiency (mpg)", y = "Weight (tons)",
       title = "This plot uses Arial Narrow fonts",
       caption = "Note: Fonts must be installed separately on your system.") +
  theme_ipsum()
```

This plot uses Arial Narrow fonts
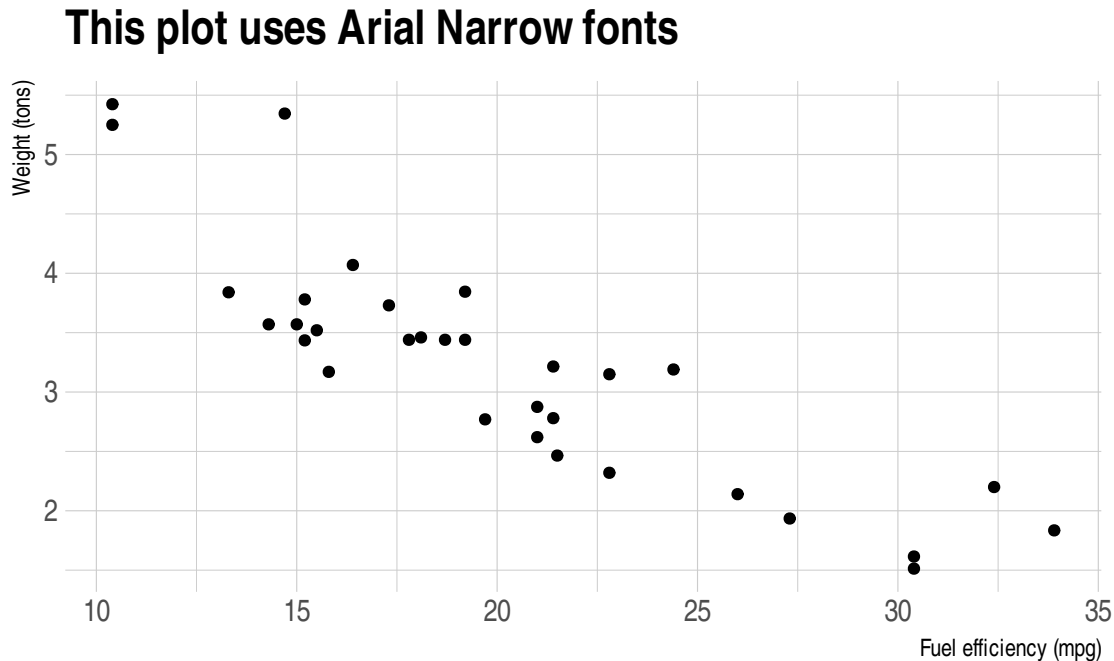
Note: Fonts must be installed separately on your system.

Ignore interactive content when exporting to PDF

In general, this template tries to do a good job of automatically handling (i.e. ignoring) interactive content when exporting to PDF. A notable exception is with embedded interactive content like external GIFs. In this case, rather than typing the usual, say, `![](mind-blown.gif)` directly in the Rmd file, you should include the figure with `knitr::include_graphics` in an R chunk. This will allow you to control whether it renders, conditional on output format. For example, the following chunk will render an actual GIF when the knit target is HTML format, versus a message when that target is PDF format.

```
## Sorry, this GIF is only available in the the HTML version of the notes.
```