Tables and Fonts

Daniel Anderson

Week 8, Class 1

Data viz in the wild

Sarah Donaldson

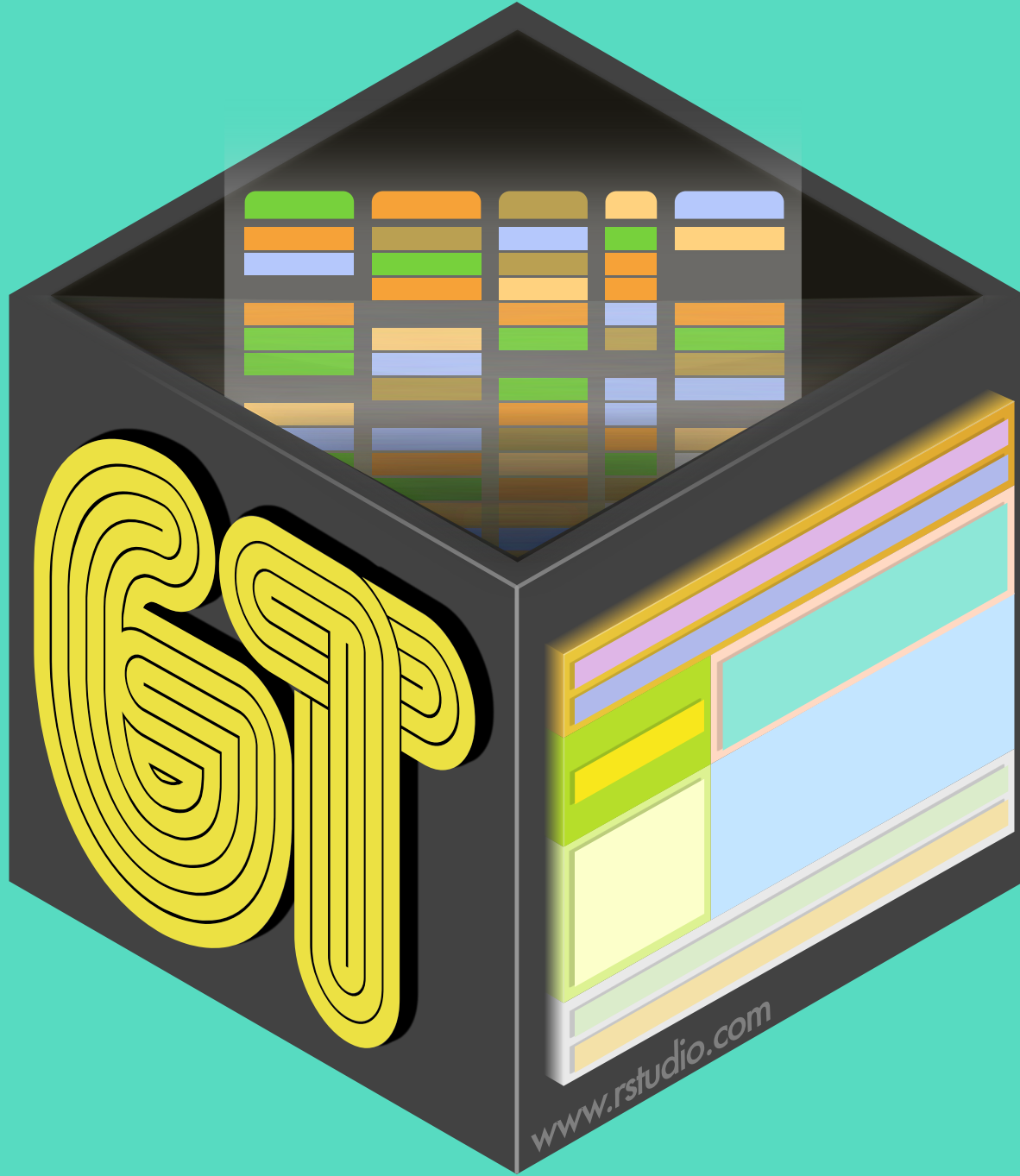Hyeonjin

Anwesha on deck

Agenda

- Tables with `gt`
- Fonts with `showtext` and/or `extrafont`

Learning objectives

- Be comfortable with the basics of `gt`
  - create a table
  - format columns
  - create spanner heads
  - etc.
- Understand how to use additional fonts (if you so choose)

Overview

- Pipe–oriented

- Beautiful tables easy

- Spanner heads/grouping used to be a total pain – not so anymore

- Renders to HTML/PDF without even thinking about it

Probably my favorite package for creating static tables, although **kableExtra** is great too.

My experience is that fewer people are generally familiar with **gt**, which is why I cover it here.

Install

```r
install.packages("gt")

# or

remotes::install_github("rstudio/gt")
```

The hard part

- Getting your data in the format you want a table in

- Utilize your `pivot_*` skills regularly

```r
library(fivethirtyeight)
flying
```

```
## # A tibble: 1,040 x 27
##    respondent_id gender age   height   children_under_18 household_incom
##            <dbl> <chr>  <ord> <ord>    <lgl>             <ord>
##  1    3436139758 <NA>   <NA>   <NA>    NA                <NA>
##  2    3434278696 Male   30-44 "6'3\"" TRUE              <NA>
##  3    3434275578 Male   30-44 "5'8\"" FALSE             $100,000 - $149
##  4    3434268208 Male   30-44 "5'11\""FALSE             $0 - $24,999
##  5    3434250245 Male   30-44 "5'7\"" FALSE             $50,000 - $99,9
##  6    3434245875 Male   30-44 "5'9\"" TRUE              $25,000 - $49,9
##  7    3434235351 Male   30-44 "6'2\"" TRUE              <NA>
##  8    3434218031 Male   30-44 "6'0\"" TRUE              $0 - $24,999
##  9    3434213681 <NA>   <NA>   "6'0\"" TRUE              <NA>
## 10    3434172894 Male   30-44 "5'6\"" FALSE             $0 - $24,999
## # … with 1,030 more rows, and 21 more variables: education <ord>, locati
## #   frequency <ord>, recline_frequency <ord>, recline_obligation <lgl>,
```

```
flying %>%
  count(gender, age, recline_frequency)
```

```
## # A tibble: 53 x 4
##    gender age   recline_frequency      n
##    <chr>  <ord> <ord>              <int>
##  1 Female 18-29 Never                 24
##  2 Female 18-29 Once in a while       36
##  3 Female 18-29 About half the time   10
##  4 Female 18-29 Usually               13
##  5 Female 18-29 Always                10
##  6 Female 18-29 <NA>                  19
##  7 Female 30-44 Never                 21
##  8 Female 30-44 Once in a while       25
##  9 Female 30-44 About half the time   22
## 10 Female 30-44 Usually               22
## # … with 43 more rows
```

```
smry <- flying %>%
  count(gender, age, recline_frequency) %>%
  drop_na(age,recline_frequency) %>%
  pivot_wider(names_from = "age",
              values_from = "n")
smry
```

```
## # A tibble: 10 x 6
##    gender recline_frequency   `18-29` `30-44` `45-60` `> 60`
##    <chr>  <ord>                 <int>   <int>   <int>  <int>
##  1 Female Never                    24      21      19     23
##  2 Female Once in a while          36      25      30     36
##  3 Female About half the time      10      22      18     17
##  4 Female Usually                  13      22      26     28
##  5 Female Always                   10      21      29     12
##  6 Male   Never                    24      17      20     18
##  7 Male   Once in a while          19      39      40     29
##  8 Male   About half the time      11      11      16     11
##  9 Male   Usually                  14      30      15     27
## 10 Male   Always                   11      14      21     14
```

Turn into table

```
library(gt)
smry %>%
  gt()
```

Disclaimer: these all look slightly different on the slides

gender	recline_frequency	18–29	30–44	45–60	> 60
Female	Never	24	21	19	23
Female	Once in a while	36	25	30	36
Female	About half the time	10	22	18	17
Female	Usually	13	22	26	28
Female	Always	10	21	29	12
Male	Never	24	17	20	18
Male	Once in a while	19	39	40	29
Male	About half the time	11	11	16	11
Male	Usually	14	30	15	27
Male	Always	11	14	21	14

Add gender as a grouping variable

```
smry %>%
  group_by(gender) %>%
  gt()
```

recline_frequency	18–29	30–44	45–60	> 60
Female				
Never	24	21	19	23
Once in a while	36	25	30	36
About half the time	10	22	18	17
Usually	13	22	26	28
Always	10	21	29	12
Male				
Never	24	17	20	18
Once in a while	19	39	40	29
About half the time	11	11	16	11
Usually	14	30	15	27
Always	11	14	21	14

This is an example of a table that looks better with the default CSS

Add a spanner head

```
smry %>%
  group_by(gender) %>%
  gt() %>%
  tab_spanner(
    label = "Age Range",
    columns = vars(`18-29`, `30-44`, `45-60`, `> 60`)
  )
```

recline_frequency	Age Range			
	18–29	30–44	45–60	> 60
Female				
Never	24	21	19	23
Once in a while	36	25	30	36
About half the time	10	22	18	17
Usually	13	22	26	28
Always	10	21	29	12
Male				
Never	24	17	20	18
Once in a while	19	39	40	29
About half the time	11	11	16	11
Usually	14	30	15	27
Always	11	14	21	14

Change column names

```
smry %>%
  group_by(gender) %>%
  gt() %>%
  tab_spanner(
    label = "Age Range",
    columns = vars(`18-29`, `30-44`, `45-60`, `> 60`)
  ) %>%
  cols_label(recline_frequency = "Recline")
```

Recline	Age Range			
	18–29	30–44	45–60	> 60
Female				
Never	24	21	19	23
Once in a while	36	25	30	36
About half the time	10	22	18	17
Usually	13	22	26	28
Always	10	21	29	12
Male				
Never	24	17	20	18
Once in a while	19	39	40	29
About half the time	11	11	16	11
Usually	14	30	15	27
Always	11	14	21	14

Align columns

```
smry %>%
  group_by(gender) %>%
  gt() %>%
  tab_spanner(
    label = "Age Range",
    columns = vars(`18-29`, `30-44`, `45-60`, `> 60`)
  ) %>%
  cols_label(recline_frequency = "Recline") %>%
  cols_align(align = "left",
             columns = vars(recline_frequency))
```

They are already left-aligned because of the CSS, but that's not typical

Recline	Age Range			
	18–29	30–44	45–60	> 60
Female				
Never	24	21	19	23
Once in a while	36	25	30	36
About half the time	10	22	18	17
Usually	13	22	26	28
Always	10	21	29	12
Male				
Never	24	17	20	18
Once in a while	19	39	40	29
About half the time	11	11	16	11
Usually	14	30	15	27
Always	11	14	21	14

Add a title

```
smry %>%
  group_by(gender) %>%
  gt() %>%
  tab_spanner(
    label = "Age Range",
    columns = vars(`18-29`, `30-44`, `45-60`, `> 60`)
  ) %>%
  cols_label(recline_frequency = "Recline") %>%
  cols_align(align = "left",
             columns = vars(recline_frequency))
  tab_header(
    title = "Airline Passengers",
    subtitle = "Leg space is limited, what do you do?"
  )
```

Airline Passengers				
Leg space is limited, what do you do?				
Recline	**Age Range**			
	18–29	**30–44**	**45–60**	**> 60**
Female				
Never	24	21	19	23
Once in a while	36	25	30	36
About half the time	10	22	18	17
Usually	13	22	26	28
Always	10	21	29	12
Male				
Never	24	17	20	18
Once in a while	19	39	40	29
About half the time	11	11	16	11
Usually	14	30	15	27
Always	11	14	21	14

Format columns

```
smry %>%
  mutate(across(c(`18-29`, `30-44`, `45-60`, `> 60`),
                ~.x/100)) %>%
  group_by(gender) %>%
  gt() %>%
  tab_spanner(
    label = "Age Range",
    columns = vars(`18-29`, `30-44`, `45-60`, `> 60`)
  ) %>%
  fmt_percent(
    vars(`18-29`, `30-44`, `45-60`, `> 60`),
    decimals = 0
  ) %>%
  cols_label(recline_frequency = "Recline") %>%
  cols_align(align = "left",
             columns = vars(recline_frequency)) %>%
  tab_header(
    title = "Airline Passengers",
    subtitle = "Leg space is limited, what do you do?"
  )
```

Recline	Age Range			
	18–29	30–44	45–60	> 60
Airline Passengers				
Leg space is limited, what do you do?				
Female				
Never	24%	21%	19%	23%
Once in a while	36%	25%	30%	36%
About half the time	10%	22%	18%	17%
Usually	13%	22%	26%	28%
Always	10%	21%	29%	12%
Male				
Never	24%	17%	20%	18%
Once in a while	19%	39%	40%	29%
About half the time	11%	11%	16%	11%
Usually	14%	30%	15%	27%
Always	11%	14%	21%	14%

Add a source note

```r
smry %>%
  mutate(across(c(`18-29`, `30-44`, `45-60`, `> 60`),
                ~.x/100)) %>%
  group_by(gender) %>%
  gt() %>%
  tab_spanner(
    label = "Age Range",
    columns = vars(`18-29`, `30-44`, `45-60`, `> 60`)
  ) %>%
  fmt_percent(
    vars(`18-29`, `30-44`, `45-60`, `> 60`),
    decimals = 0
  ) %>%
  cols_label(recline_frequency = "Recline") %>%
  cols_align(align = "left",
             columns = vars(recline_frequency)) %>%
  tab_header(
    title = "Airline Passengers",
    subtitle = "Leg space is limited, what do you do?"
  ) %>%
  tab_source_note(
    source_note = md("Data from [fivethirtyeight](https://fiveth
```

Airline Passengers				
Leg space is limited, what do you do?				
Recline	Age Range			
	18–29	30–44	45–60	> 60
Female				
Never	24%	21%	19%	23%
Once in a while	36%	25%	30%	36%
About half the time	10%	22%	18%	17%
Usually	13%	22%	26%	28%
Always	10%	21%	29%	12%
Male				
Never	24%	17%	20%	18%
Once in a while	19%	39%	40%	29%
About half the time	11%	11%	16%	11%
Usually	14%	30%	15%	27%
Always	11%	14%	21%	14%
Data from fivethirtyeight				

Color cells

```
... %>%
  data_color(
    vars(`18-29`, `30-44`, `45-60`, `> 60`),
    colors = scales::col_numeric(
      palette = c("#FFFFFF", "#FF0000"),
      domain = NULL
    )
  ) %>%
  ...
```

Recline	Age Range			
	18–29	30–44	45–60	> 60
Female				
Never	24%	21%	19%	23%
Once in a while	36%	25%	30%	36%
About half the time	10%	22%	18%	17%
Usually	13%	22%	26%	28%
Always	10%	21%	29%	12%
Male				
Never	24%	17%	20%	18%
Once in a while	19%	39%	40%	29%
About half the time	11%	11%	16%	11%
Usually	14%	30%	15%	27%
Always	11%	14%	21%	14%
Data from fivethirtyeight				

Airline Passengers

Leg space is limited, what do you do?

What else?

- Lots more it can do, and lots more in development
- See the website

Thomas Mock does a lot of great work with tables and often has tutorials showing your how to go further (e.g., see here and here and here).

A few other table options

kableExtra

A few quick examples

Make sure to specify `results = "asis"` in your chunk options.

```r
library(knitr)
library(kableExtra)
dt <- mtcars[1:5, 1:6]
kable(dt) %>%
  kable_styling("striped") %>%
  column_spec(5:7, bold = TRUE
```

	mpg	cyl	disp	hp	drat	wt
Mazda RX4	21.0	6	160	**110**	**3.90**	**2.620**
Mazda RX4 Wag	21.0	6	160	**110**	**3.90**	**2.875**
Datsun 710	22.8	4	108	**93**	**3.85**	**2.320**
Hornet 4 Drive	21.4	6	258	**110**	**3.08**	**3.215**
Hornet Sportabout	18.7	8	360	**175**	**3.15**	**3.440**

```
kable(dt) %>%
  kable_styling("striped") %>%
  column_spec(5:7, bold = TRUE) %>%
  row_spec(c(2, 4),
           bold = TRUE,
           color = "#EFF3F7",
           background = "#71B0DE")
```

	mpg	cyl	disp	hp	drat	wt
Mazda RX4	21.0	6	160	**110**	**3.90**	**2.620**
Mazda RX4 Wag	**21.0**	**6**	**160**	**110**	**3.90**	**2.875**
Datsun 710	22.8	4	108	**93**	**3.85**	**2.320**
Hornet 4 Drive	**21.4**	**6**	**258**	**110**	**3.08**	**3.215**
Hornet Sportabout	18.7	8	360	**175**	**3.15**	**3.440**

```r
kable(dt) %>%
  kable_styling("striped", full_width = FALSE) %>%
  pack_rows(
    "Group 1", 1, 3,
    label_row_css = "background-color: #666; color: #fff;"
    ) %>%
  pack_rows(
    "Group 2", 4, 5,
    label_row_css = "background-color: #666; color: #fff;")
```

	mpg	cyl	disp	hp	drat	wt
Group 1						
Mazda RX4	21.0	6	160	110	3.90	2.620
Mazda RX4 Wag	21.0	6	160	110	3.90	2.875
Datsun 710	22.8	4	108	93	3.85	2.320
Group 2						
Hornet 4 Drive	21.4	6	258	110	3.08	3.215
Hornet Sportabout	18.7	8	360	175	3.15	3.440

KableExtra wrapup

Many other options, please see the documentation. Works well for PDF and HTML.

What about Microsoft Word?

flextable

flextable R package

`build` `passing` `BUILD PASSING` `CRAN` `0.5.8` `downloads` `16K/month` `repo status` `Active`

The flextable package provides a framework for easily create tables for reporting and publications. Tables can be embedded within:

- R Markdown documents with support for HTML, Word and PowerPoint documents.
- Microsoft Word or PowerPoint documents.
- PDF documents with package `pagedown` (it's only HTML)

Tables can also be rendered as R plots or graphic files (png, pdf and jpeg).

🔗 Getting Started

An API is available to let R users create tables for reporting and control their formatting properties and their layout. A `flextable` object is a data.frame representation, it can be manipulated with functions that give control over:

Links

Download from CRAN at
https://cloud.r-project.org/
package=flextable

Report a bug at
https://github.com/davidgohel/flextable/
issues

Visit ARDATA website at
https://www.ardata.fr

License

GPL-3

Developers

David Gohel
Author, maintainer

All authors...

reactable

Probably my favorite for interactive HTML tables – works great with **shiny** too

Many others

- huxtable
- formattable
- DT (my former favorite for shiny)
- rhandsontable

Particularly helpful for modeling

- stargazer
- pixiedust
- modelsummary

For descriptives

- gtsummary

Fonts

General advice

- Use different fonts to distinguish things

  - Specifically code
  - Consider different fonts for different heading levels, and/or to distinguish headers from the body

- Always choose a sans–serif font for code

- Explore and try – it makes a big impact on the overall look/feel (bigger than you may expect if you haven't played with fonts much before)

- Try not to get sucked into too deep of a rabbit hole

Google fonts

https://fonts.google.com

- Open source, designed for the web

- Good place to explore fonts

- Can be incorporated via the `{showtext}` package!

{showtext} example

```
devtools::install_github("yixuan/showtext")

library(showtext)

font_add_google('Monsieur La Doulaise', "mld")
font_add_google('Special Elite', "se")

showtext_auto()
quartz()

ggplot(mtcars, aes(disp, mpg)) +
  geom_point() +
  labs(title = "An amazing title",
       subtitle = "with the world's most boring dataset") +
  theme(plot.subtitle = element_text(size = 18, family = "se"),
        plot.title = element_text(size = 22, family = "mld"),
        axis.title = element_text(size = 18, family = "mld"),
        axis.text.x = element_text(size = 12, family = "se"),
        axis.text.y = element_text(size = 12, family = "se"))
```

An amazing title

with the world's most boring dataset
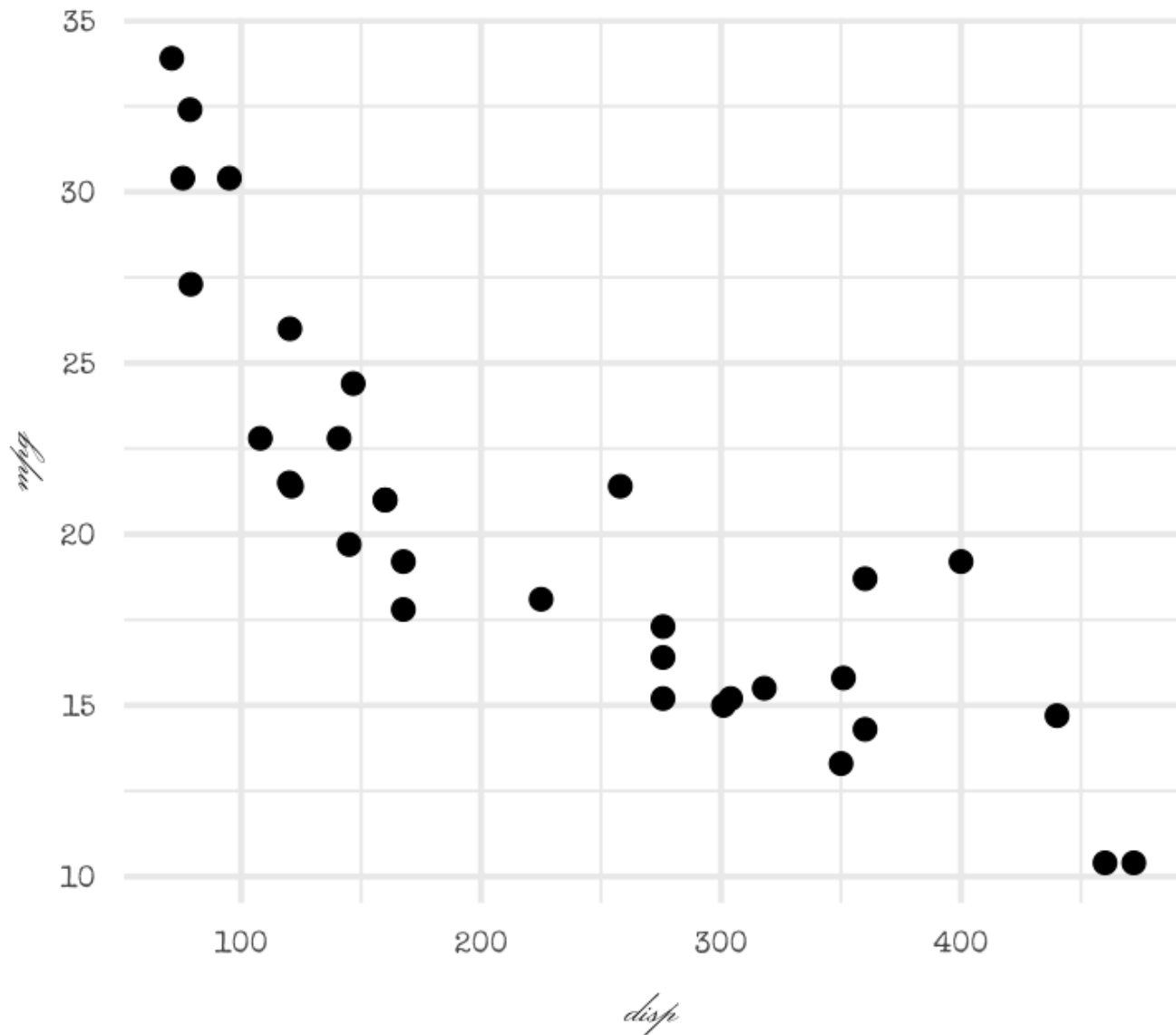
extrafont

- Primary downside – you have to have the fonts installed on your computer
- Look at the install documentation – it's pretty good and fairly comprehensive

```
library(extrafont)
# font_import() Only need to run once
fonts() # list fonts
```

```
##   [1] "Roboto Condensed"        ".SF Compact Display"
##   [3] ".SF Compact Rounded"     ".SF Compact Text"
##   [5] ".Keyboard"               ".New York"
##   [7] "System Font"             ".SF NS Mono"
##   [9] ".SF NS Rounded"          "Andale Mono"
##  [11] "Apple Braille"           "AppleMyungjo"
##  [13] "Arial Black"             "Arial"
##  [15] "Arial Narrow"            "Arial Rounded MT Bold"
##  [17] "Arial Unicode MS"        "Bodoni Ornaments"
##  [19] "Bodoni 72 Smallcaps"     ""
##  [21] "Brush Script MT"         "Comic Sans MS"
##  [23] "Courier New"             "DIN Alternate"
```

```
ggplot(mtcars, aes(wt, mpg)) +
  geom_point() +
  labs(title = "Fuel Efficiency of 32 Cars",
       x = "Weight (x1000 lb)",
       y = "Miles per Gallon") +
  theme(text = element_text(family = "Luminari", size = 30))
```
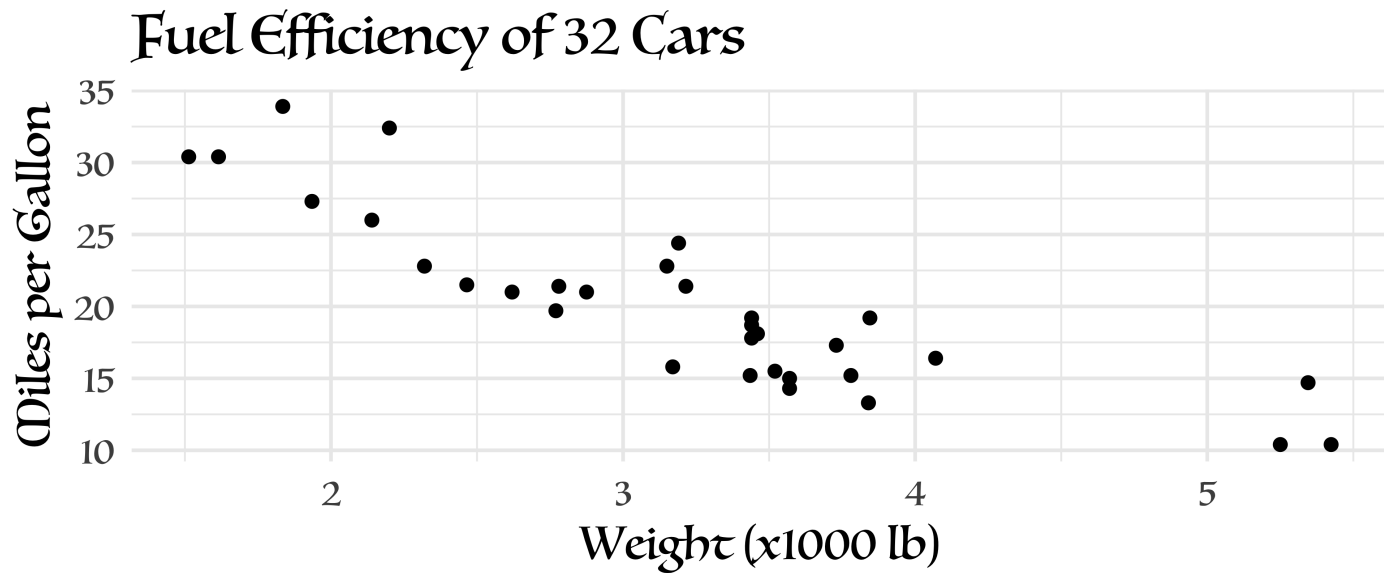
Why fonts matter

A few examples of epic fails

h/t Will Chase

Megaflicks

Quick aside

Change the font of your R Markdown!

Create a new text file, import the text there, write a tiny bit of css.

```css
@import url('https://fonts.googleapis.com/css?family=Akronim&disp

body {
  font-family: 'Akronim', cursive;
}
```

Save this in a new file in the same directory.

Modify Rmd YAML

You need to have the CSS file in the same directory as your Rmd, then reference it with

```
output:
  html_document:
    css: custom.css
```

Here I'm assuming the file you saved before is called "custom.css".

Render!

Alternative approach

It's good to think more about CSS and how it links with everything HTML. We'll talk about this a bit more next week. But, if you only have a little bit of CSS you need to write, you can do it right within RMarkdown!

I actually did this for these slides to make the tables a bit smaller!

```
24
25 ```{css echo = FALSE}
26 table {
27    font-size: 1rem;
28 }
29 ```
30
```

Resource for learning more

- I'm not an expert on fonts. I have mostly just picked what looks nice to me.

Best I've heard of is practical typography

Identify fonts

Use others work to help you – I found the font for these slides from a different theme that I liked.

Use google chrome's developer tools to help!

Next time

Create your own blog!