

# APPLICATIONS



OF DATA SCIENCE

# Visualizing in the Tidyverse

## Applications of Data Science - Class 4

Giora Simchoni

[gsimchoni@gmail.com](mailto:gsimchoni@gmail.com) and add #dsapps in subject

Stat. and OR Department, TAU

2021-01-19

APPLICATIONS



OF DATA SCIENCE

# Welcome to ggplot2

Heavily inspired by [datascienceinabox.org](http://datascienceinabox.org) / @minebocek

APPLICATIONS



OF DATA SCIENCE

# ggplot()

- **ggplot2** is tidyverse's data visualization package
- The **gg** in "ggplot2" stands for Grammar of Graphics
- It is inspired by the book [Grammar of Graphics](#) by Leland Wilkinson
- A grammar of graphics is a tool that enables us to concisely describe the components of a graphic



# Remember the Force, Luke!

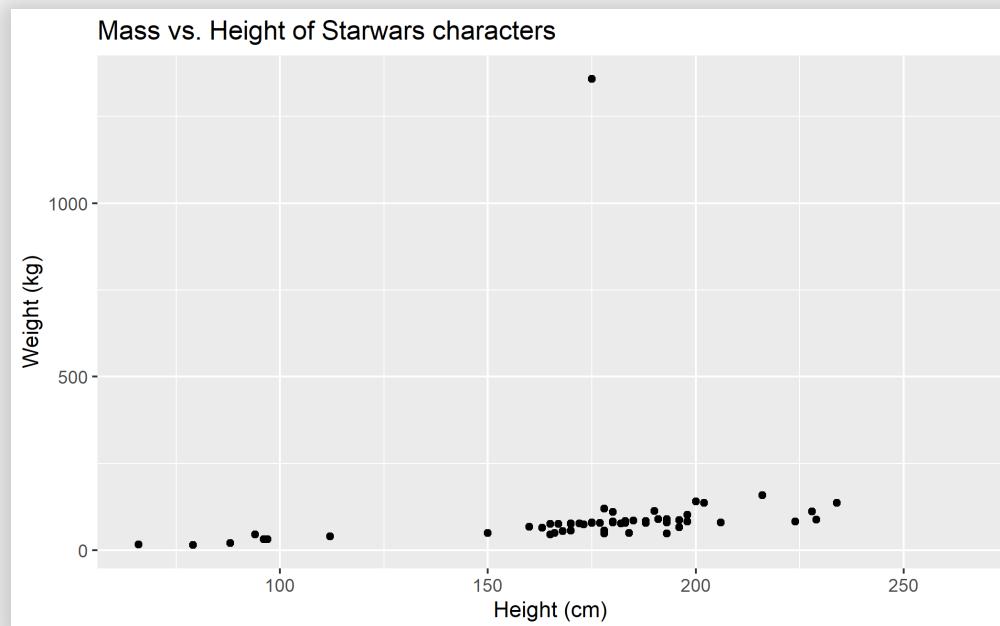
```
sw_tables <- read_rds("../data/sw_tables.rds")
characters <- sw_tables$characters
planets <- sw_tables$planets
films <- sw_tables$films

qglimpse(characters)
```

What are the functions doing the plotting? What is the dataset being plotted? Which variable is on the x-axis and which variable is on the y-axis? What does the warning mean?

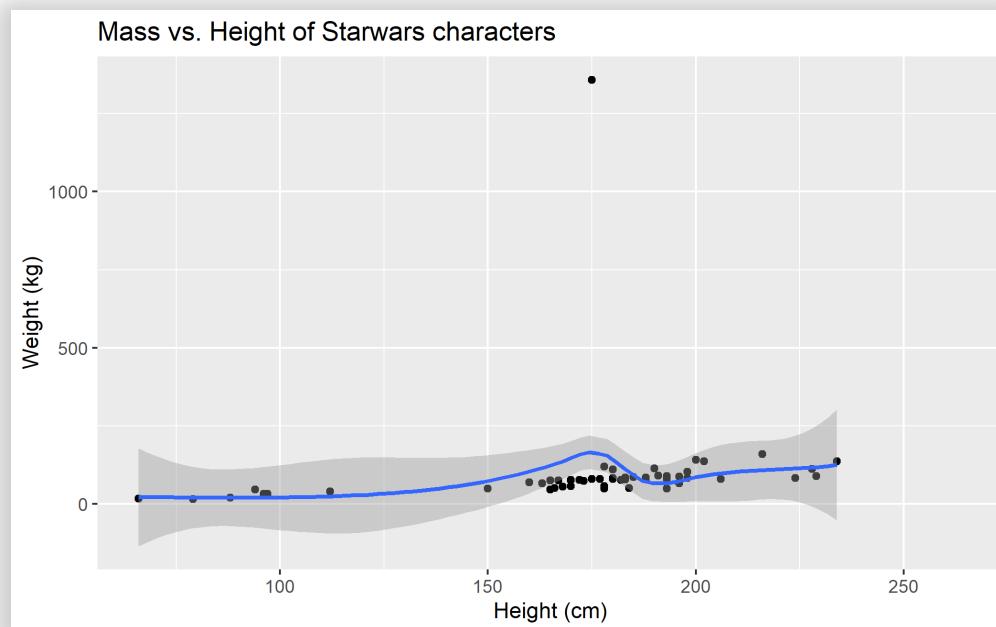
```
ggplot(data = characters, mapping = aes(x = height, y = mass)) +  
  geom_point() +  
  labs(title = "Mass vs. Height of Starwars characters",  
       x = "Height (cm)", y = "Weight (kg)")
```

`## Warning: Removed 36 rows containing missing values (geom_point).`



What does `geom_smooth()` do? What else changed between the previous plot and this one?

```
ggplot(data = characters, mapping = aes(x = height, y = mass)) +  
  geom_point() +  
  geom_smooth() +  
  labs(title = "Mass vs. Height of Starwars characters",  
       x = "Height (cm)", y = "Weight (kg)")
```



# Who dis!

- `ggplot()` is the main function in `ggplot2` and plots are constructed in layers
- The structure of the code for plots can often be summarized as

```
ggplot +  
  geom_xxx
```

or, more precisely

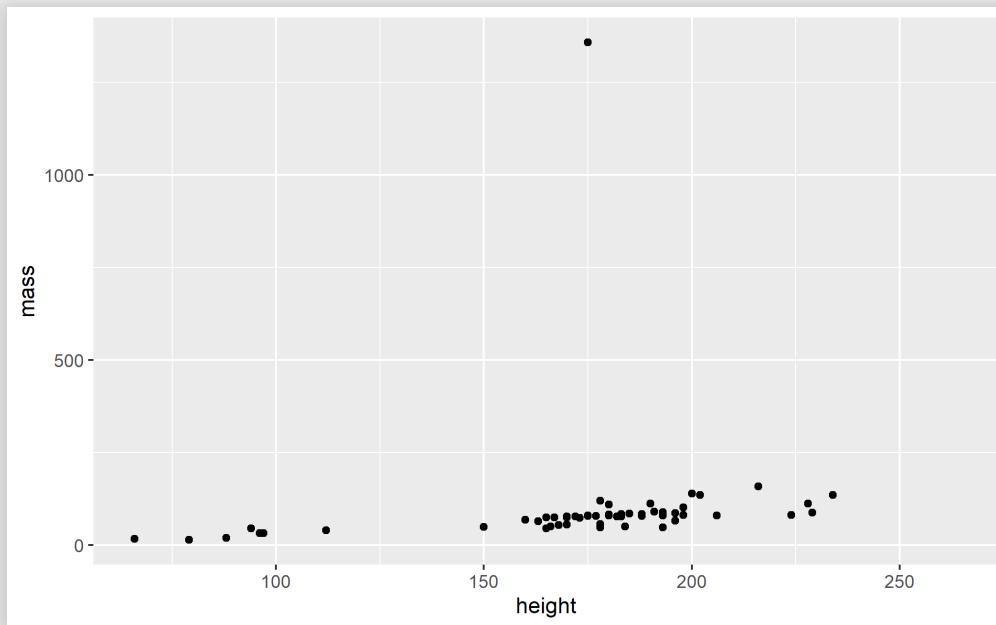
```
ggplot(data = [dataset], mapping = aes(x = [x-variable], y = [y-varaible]),  
       geom_xxx() +  
       other options)
```

- To use `ggplot2` functions, first load `tidyverse`

```
library(tidyverse)
```

# Mass vs. Height

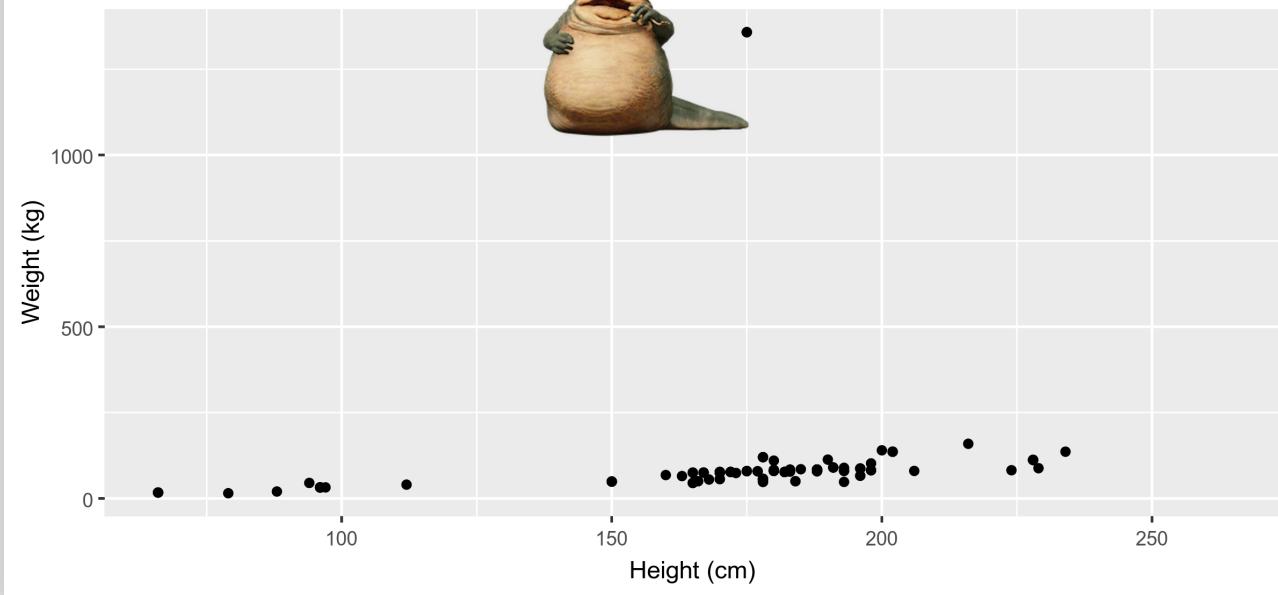
```
ggplot(characters, aes(height, mass)) + geom_point()
```



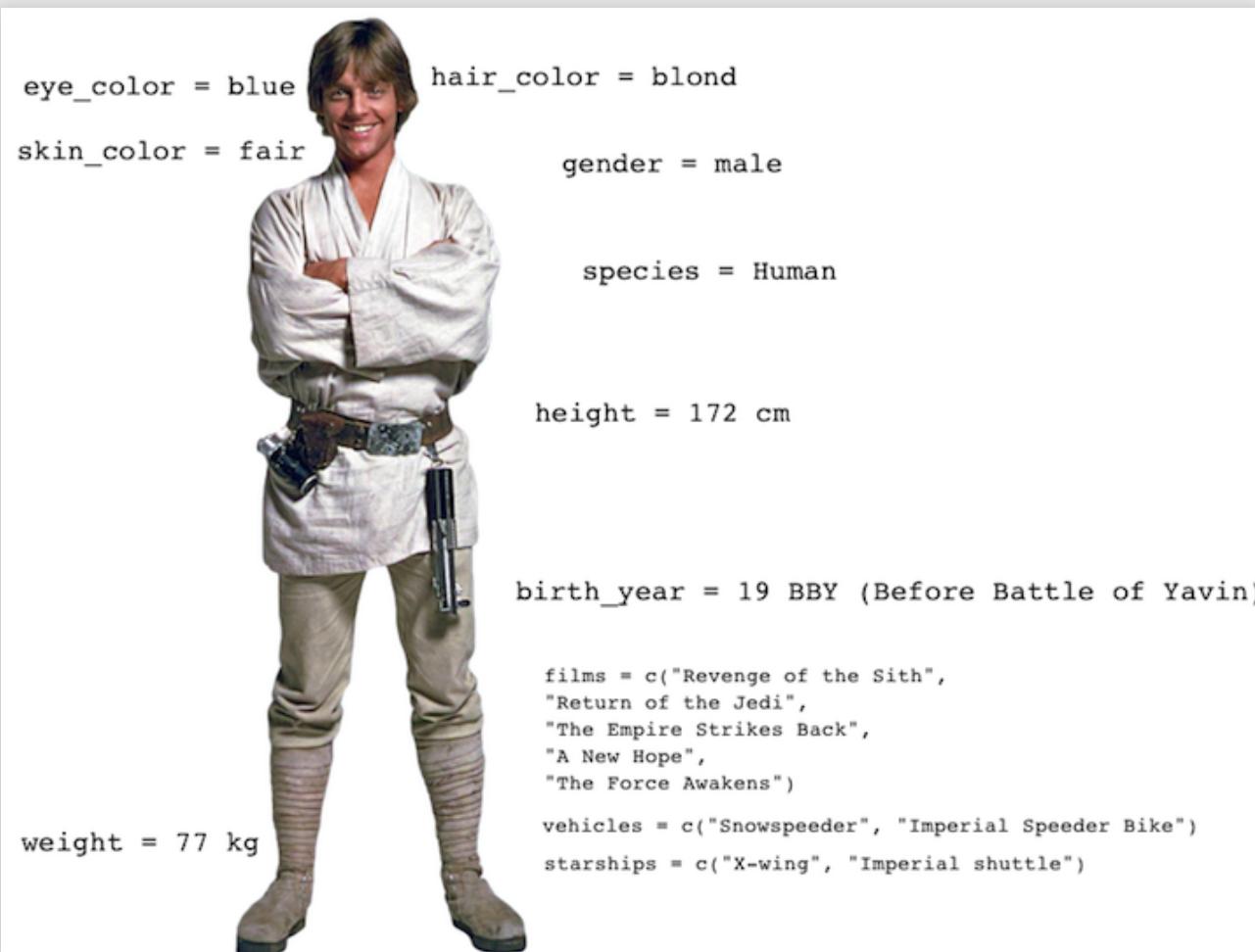
How would you describe this relationship? What other variables would help us understand data points that don't follow the overall trend? Who is the not so tall but really chubby character?

# Jabba!

Mass vs. Height of Starwars characters



# But seriously, Luke



# Additional variables

We can map additional variables to various features of the plot:

- aesthetics
  - shape
  - colour
  - size
  - alpha (transparency)
- faceting: small multiples displaying different subsets

# Aesthetics

APPLICATIONS



OF DATA SCIENCE

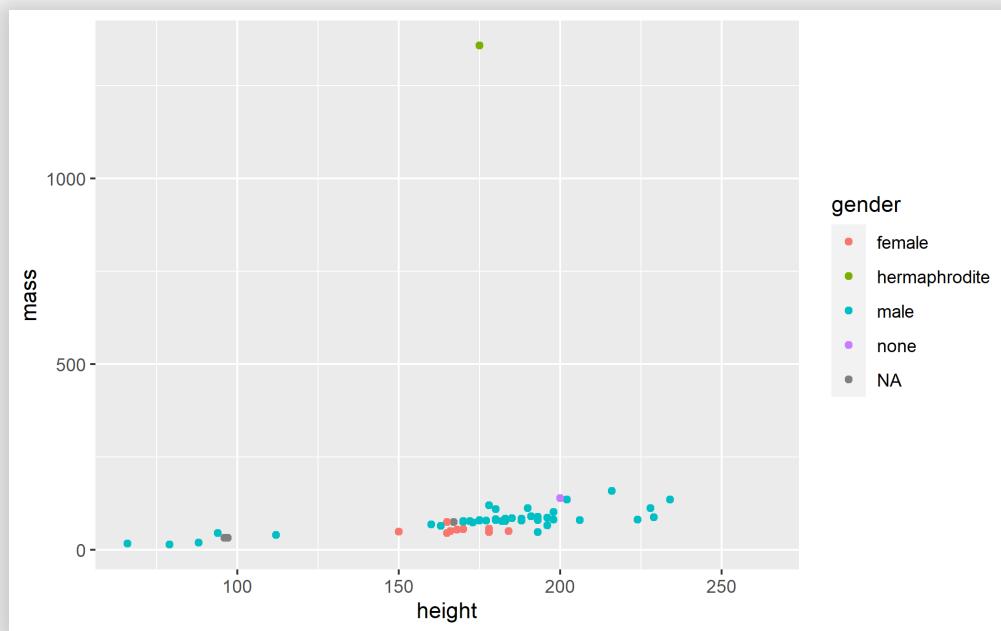
# Aesthetics options

Visual characteristics of plotting characters that can be **mapped to a specific variable** in the data are

- color
- fill
- size
- shape
- alpha (transparency)

# Mass vs. Height + Gender

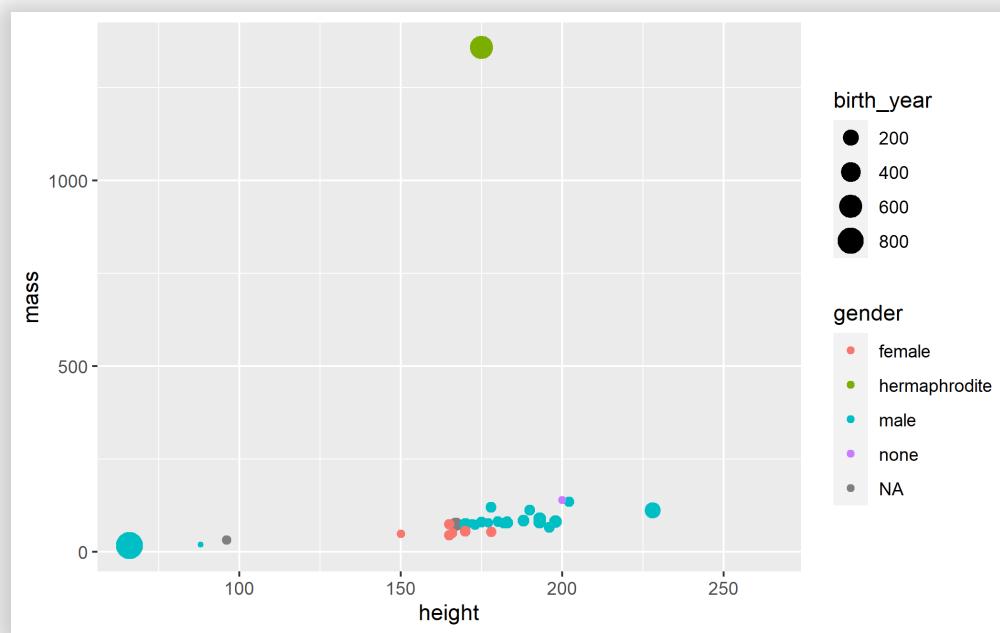
```
ggplot(data = characters,  
       mapping = aes(x = height, y = mass, color = gender)) +  
  geom_point()
```



# Mass vs. Height + Gender + Birth Year

Let's map the size to birth\_year:

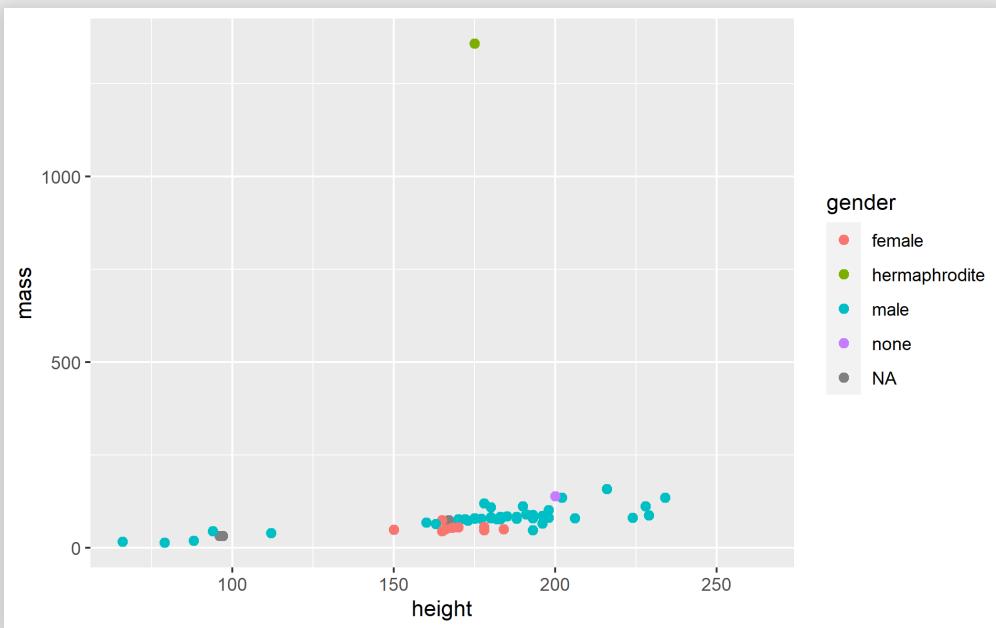
```
ggplot(data = characters,  
       mapping = aes(x = height, y = mass, color = gender,  
                      size = birth_year)) +  
  geom_point()
```



# Mass vs. Height + Gender

Let's now increase the size of all points **not** based on the values of a variable in the data:

```
ggplot(data = characters,  
       mapping = aes(x = height, y = mass, color = gender)) +  
  geom_point(size = 2)
```



# Aesthetics Summary

- Continuous variables are measured on a continuous scale
- Discrete variables are measured (or often counted) on a discrete scale

aesthetics	discrete	continuous
color	rainbow of colors	gradient
size	discrete steps	linear mapping between radius and value
shape	different shape for each	shouldn't (and doesn't) work

- Use aesthetics for mapping features of a plot to a variable, define the features in the geom for customization **not** mapped to a variable

# Faceting

APPLICATIONS

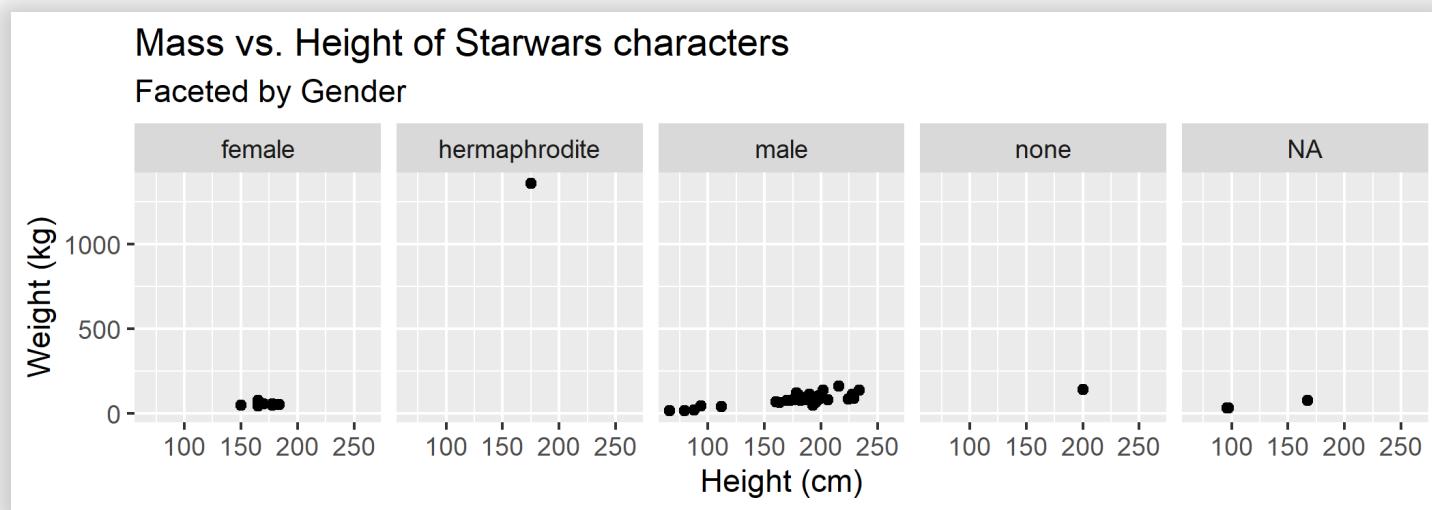


OF DATA SCIENCE

# Faceting options

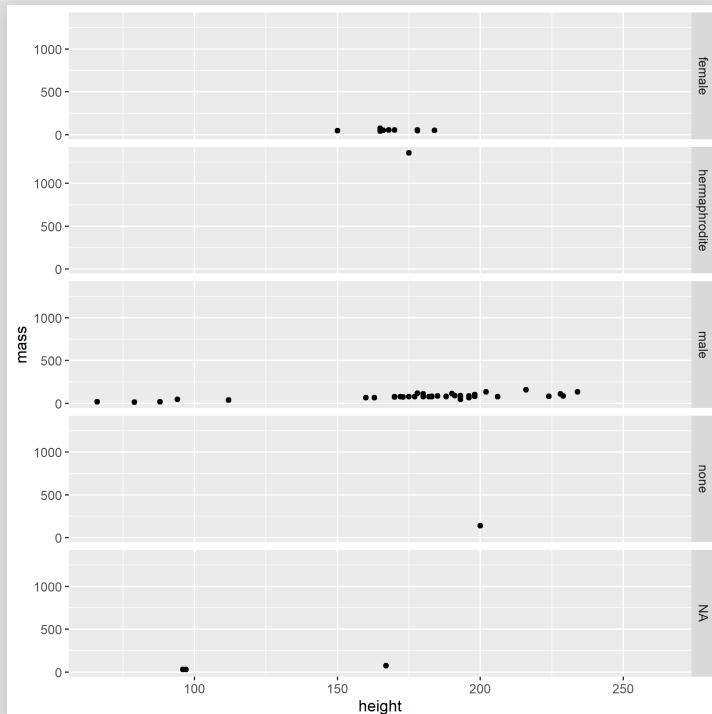
- Smaller plots that display different subsets of the data
- Useful for exploring conditional relationships and large data

```
ggplot(data = characters, mapping = aes(x = height, y = mass)) +  
  facet_grid(. ~ gender) +  
  geom_point() +  
  labs(title = "Mass vs. Height of Starwars characters",  
       subtitle = "Faceted by Gender",  
       x = "Height (cm)", y = "Weight (kg)")
```

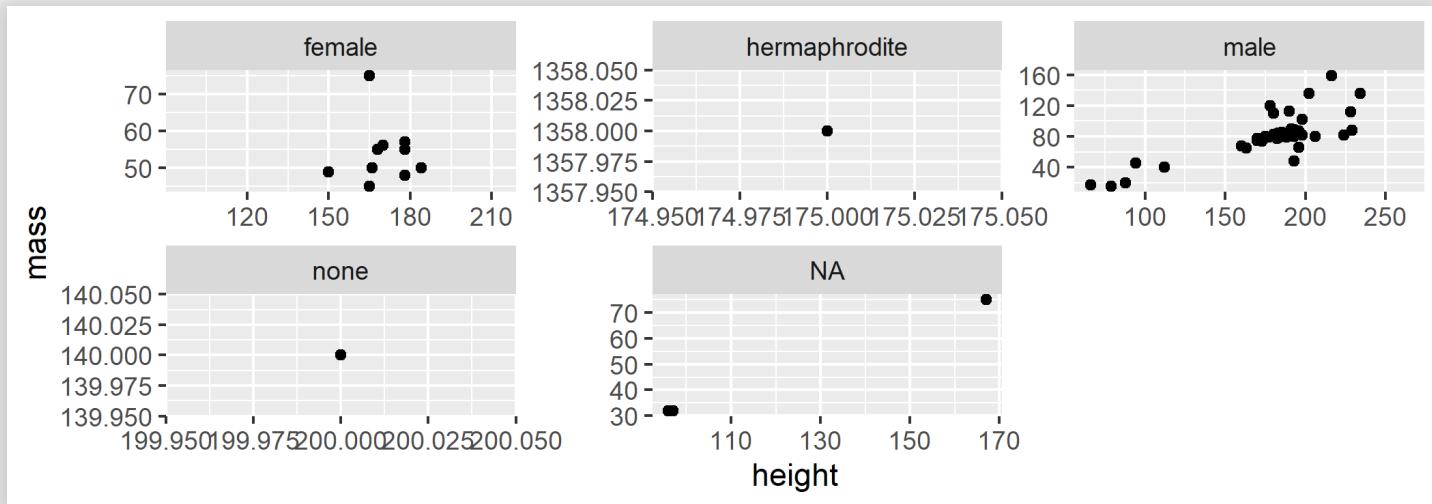


# Many ways to facet

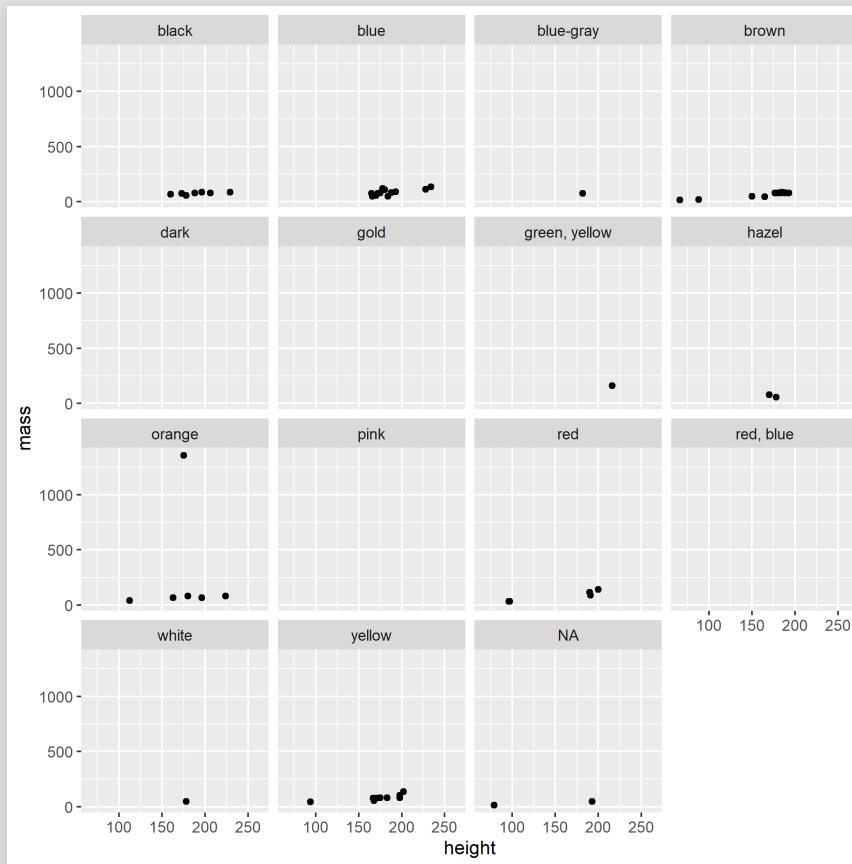
```
ggplot(data = characters, mapping = aes(x = height, y = mass)) +  
  geom_point() +  
  facet_grid(gender ~ .)
```



```
ggplot(data = characters, mapping = aes(x = height, y = mass)) +  
  geom_point() +  
  facet_wrap(. ~ gender, scales = "free")
```



```
ggplot(data = characters, mapping = aes(x = height, y = mass)) +  
  geom_point() +  
  facet_wrap(~ eye_color)
```



# Facet summary

- `facet_grid()`:
  - 2d grid
  - `rows ~ cols`
  - use `.` for no split
- `facet_wrap()`: 1d ribbon wrapped into 2d

# Take Control

APPLICATIONS



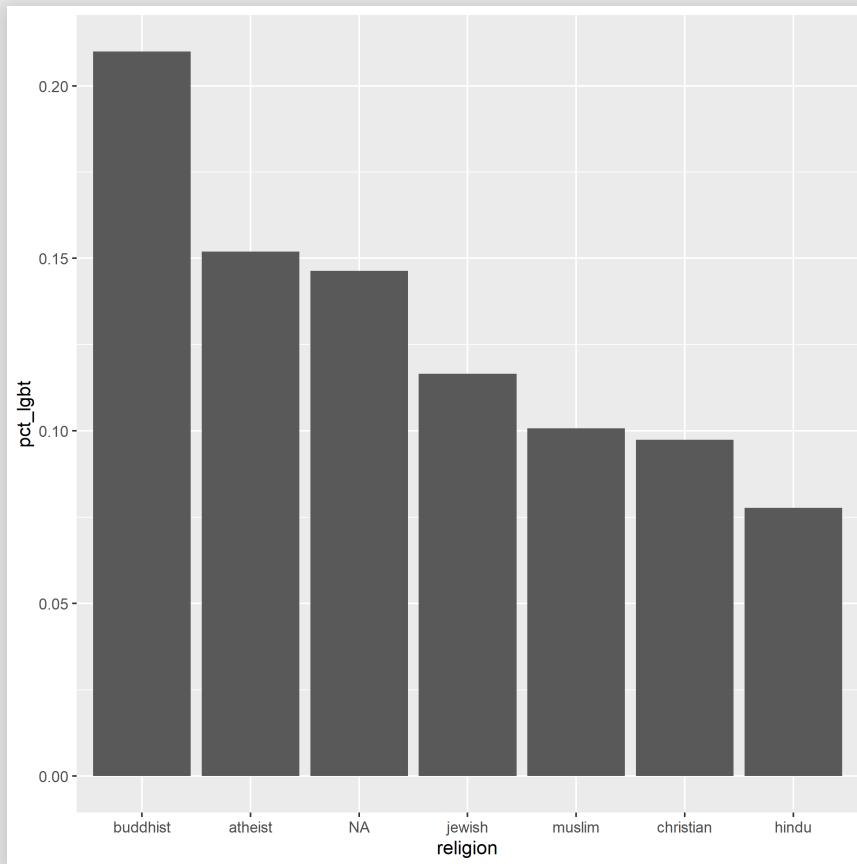
OF DATA SCIENCE

```
okcupid_lgbt_relig <- okcupid %>%
  group_by(religion2) %>%
  summarise(pct_lgbt = mean(orientation %in% c("gay", "bisexual")))
  mutate(religion = fct_reorder(religion2, desc(pct_lgbt)))

okcupid_lgbt_relig
```

```
## # A tibble: 7 x 3
##   religion2 pct_lgbt religion
##   <chr>        <dbl> <fct>
## 1 atheist      0.152 atheist
## 2 buddhist     0.210 buddhist
## 3 christian    0.0974 christian
## 4 hindu        0.0778 hindu
## 5 jewish       0.117 jewish
## 6 muslim        0.101 muslim
## 7 NA            0.146 NA
```

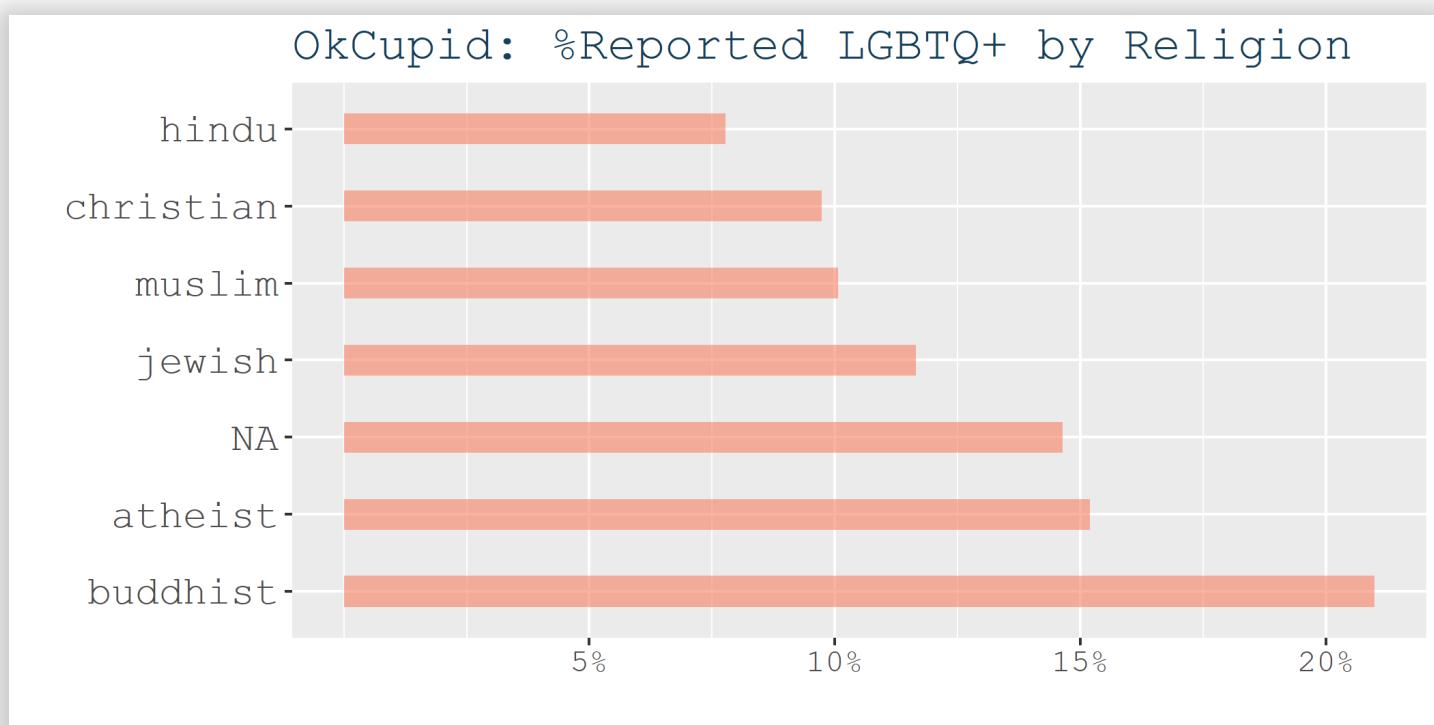
```
okcupid_lgbt_relig %>%  
  ggplot(aes( religion, pct_lgbt)) +  
  geom_bar(stat="identity")
```



# Pimp my plot

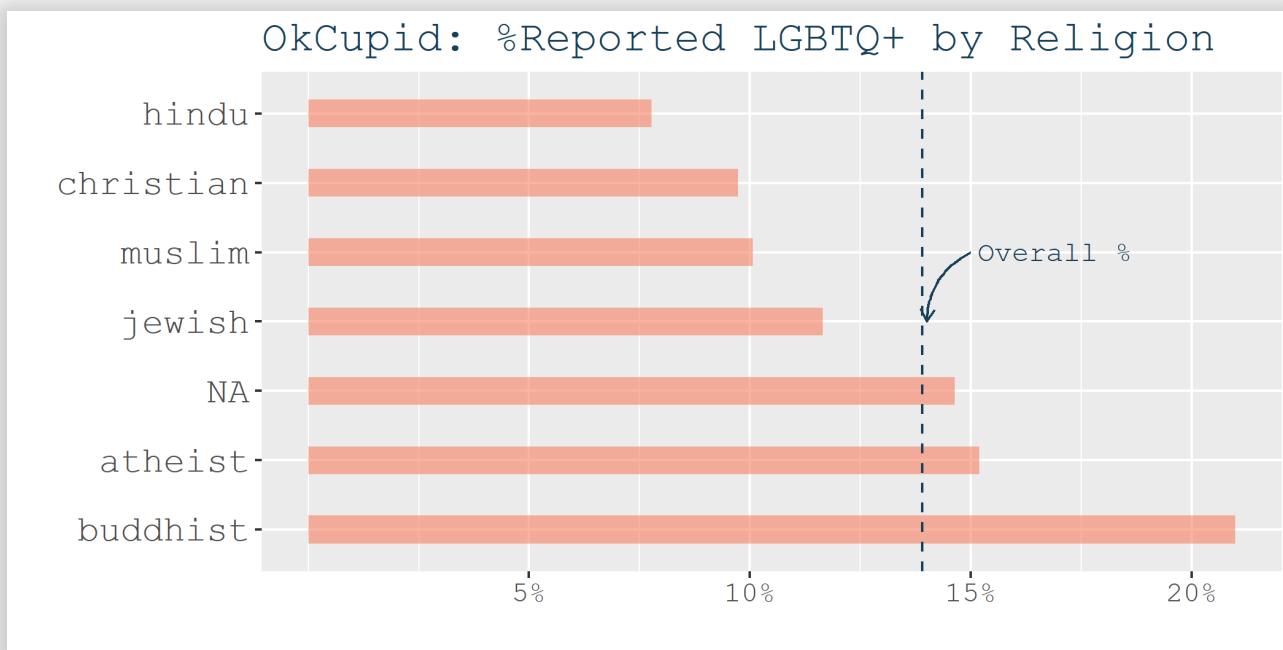
```
p <- ggplot(okcupid_lgbt_relig, aes(religion, pct_lgbt)) +
  geom_bar(stat="identity", fill="#f68060", alpha=.6, width=.4) +
  labs(x = "", y = "",
       title = "OkCupid: %Reported LGBTQ+ by Religion") +
  theme(text = element_text(size = 14, color = "#163f59", family =
    axis.text.y = element_text(size = 14)) +
    scale_y_continuous(breaks = seq(0.05, 0.2, 0.05),
                       labels=scales::percent_format(accuracy = 1)))
  coord_flip()
p
```

# Pimp my plot



# Annotate, Add lines

```
p + geom_hline(yintercept = 0.139, lty = 2, color = "#163f59") +  
  annotate("text", x = 5, y = 0.139 + 0.03, label = "Overall %",  
          color = "#163f59", family = "mono") +  
  annotate(geom = "curve", x = 5, y = 0.15, xend = 4, yend = 0.139,  
          curvature = .3, arrow = arrow(length = unit(2, "mm"))), color =
```



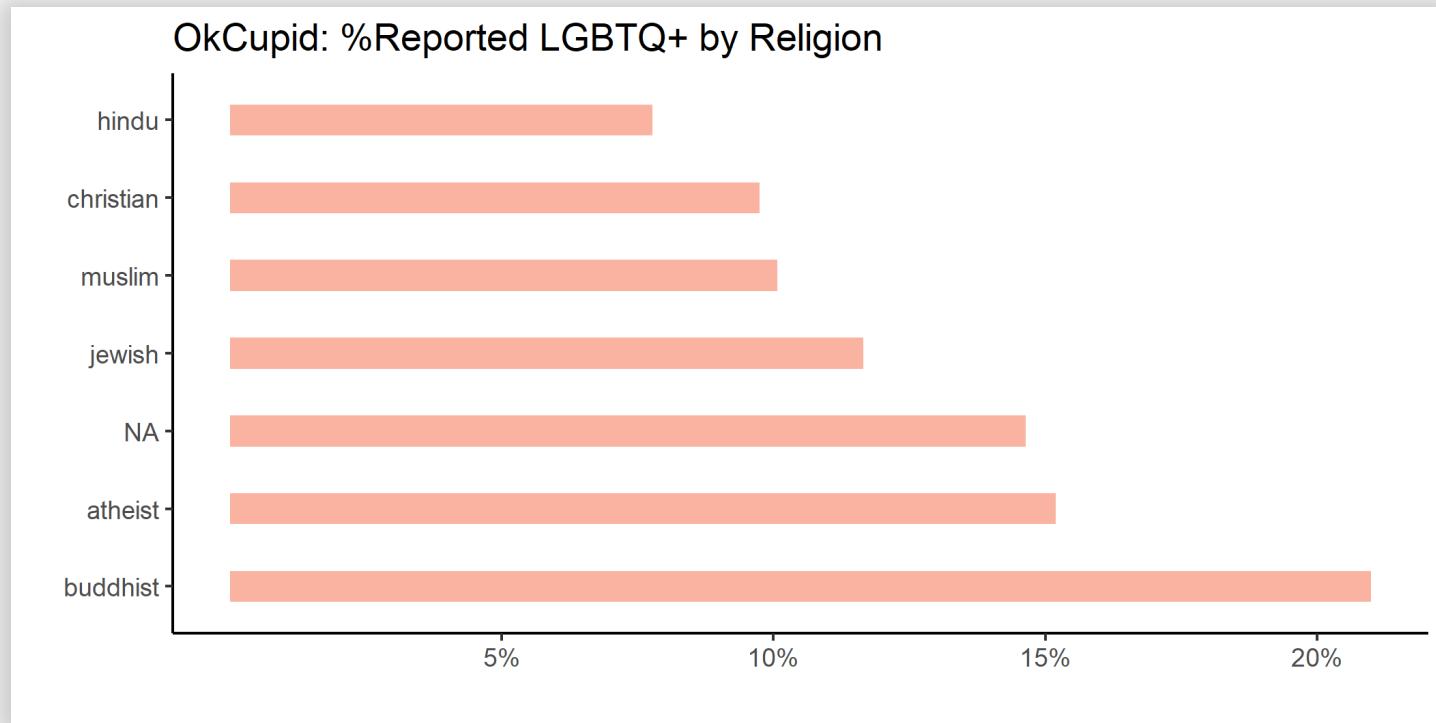
# Themes

APPLICATIONS

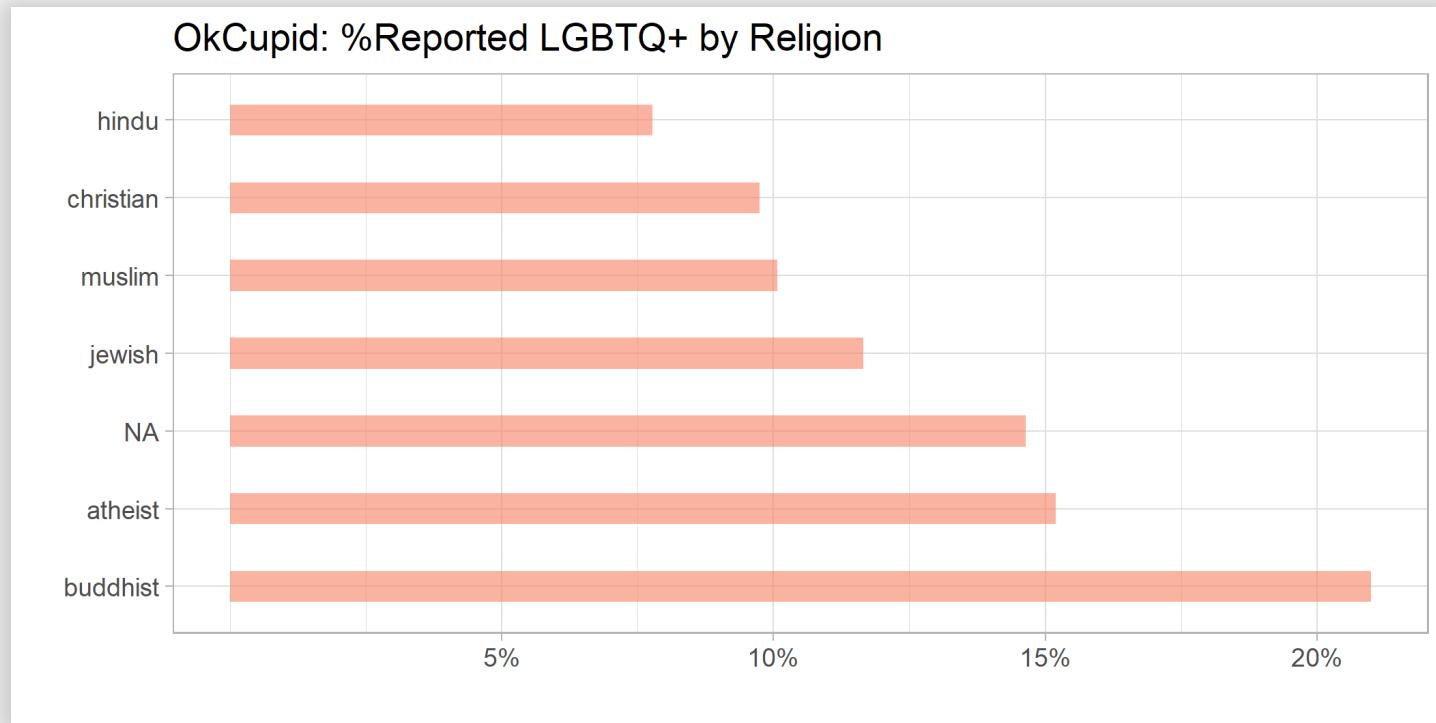


OF DATA SCIENCE

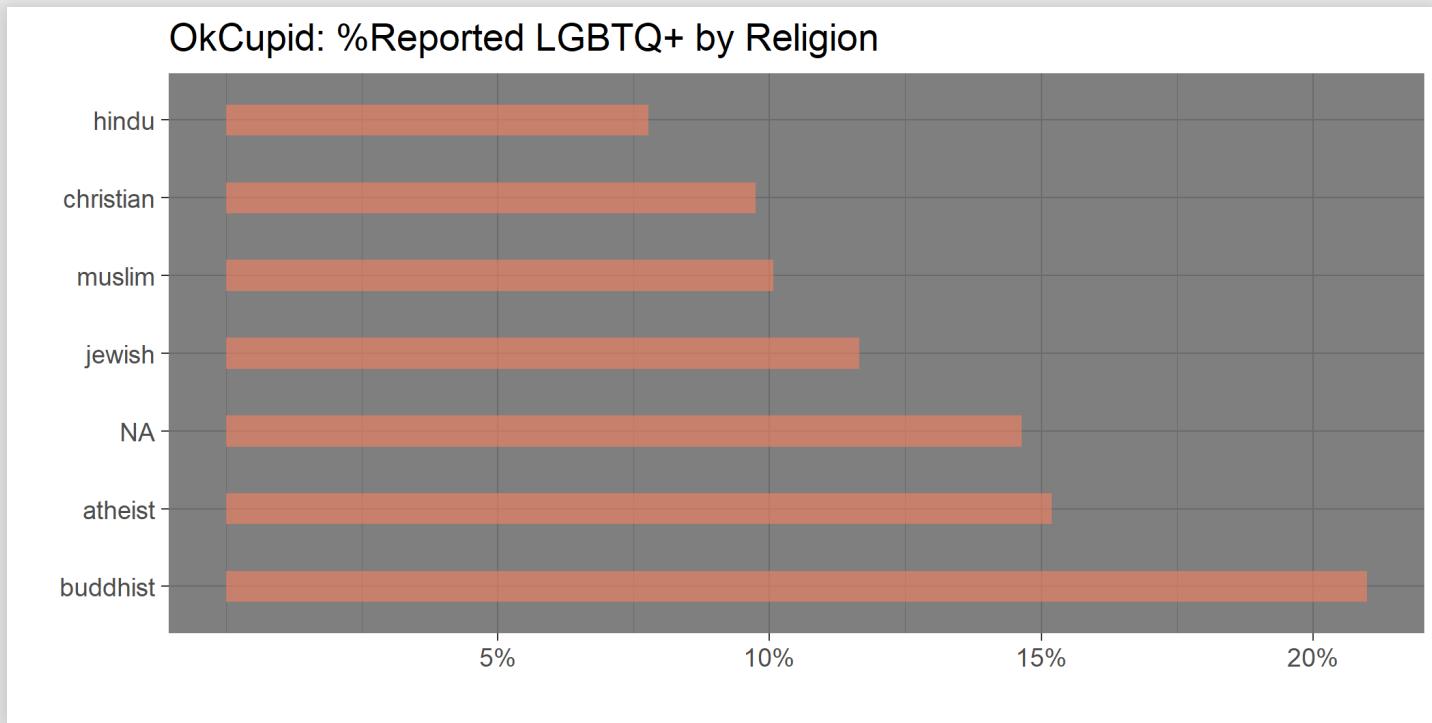
```
p + theme_classic()
```



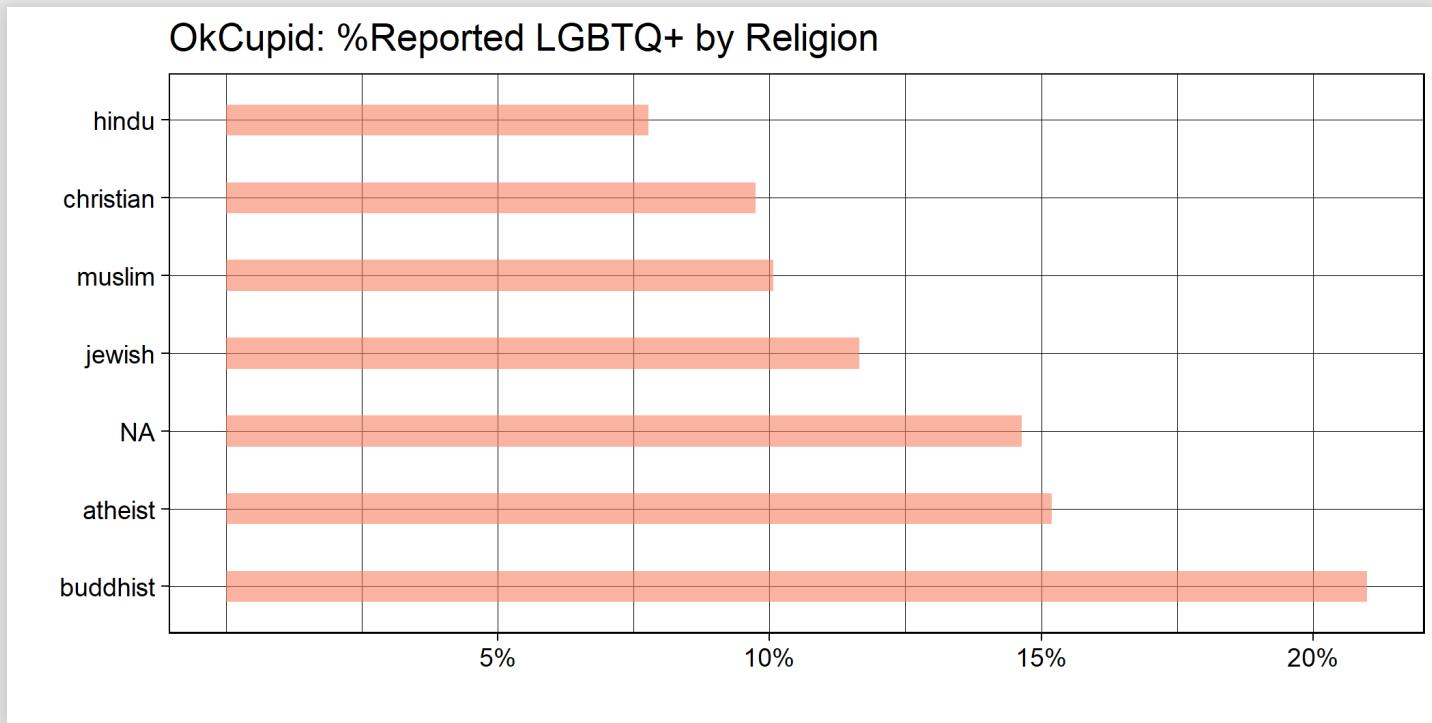
```
p + theme_light()
```



```
p + theme_dark()
```

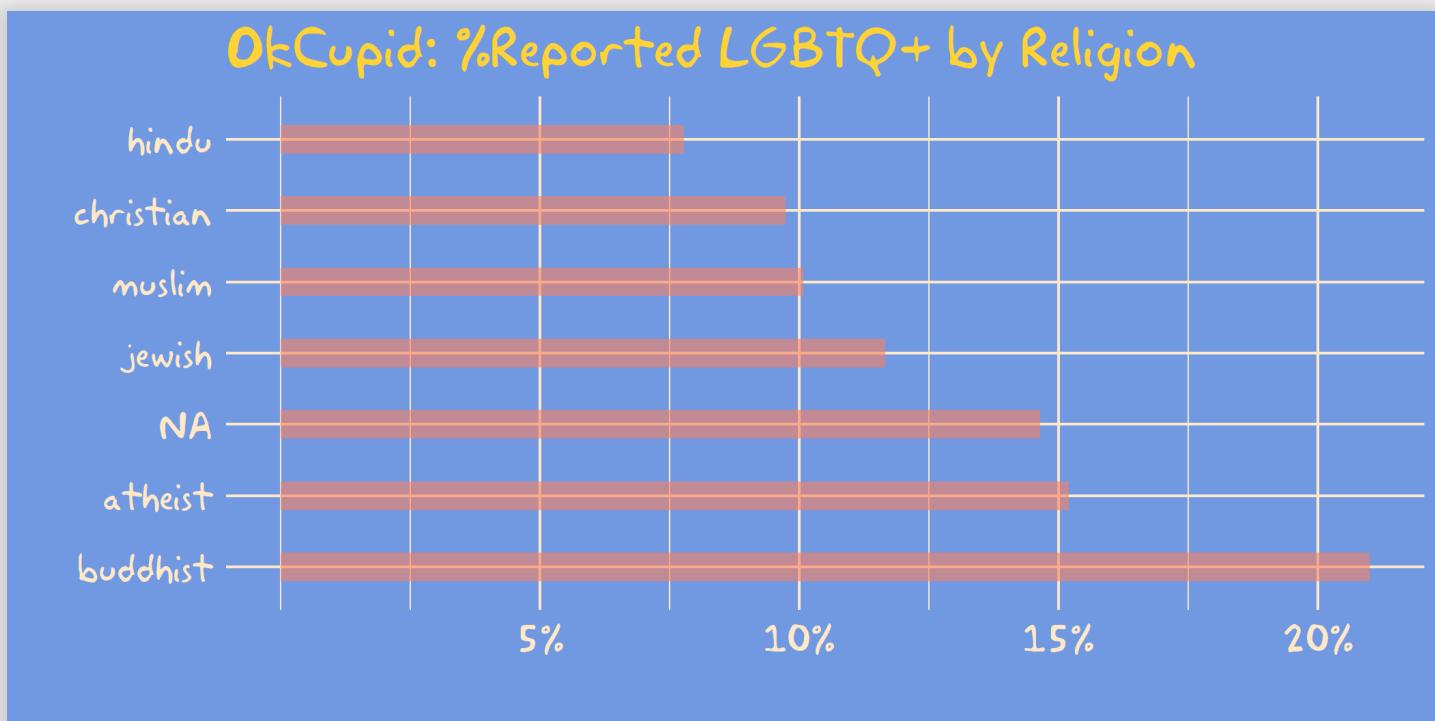


```
p + theme_linedraw()
```



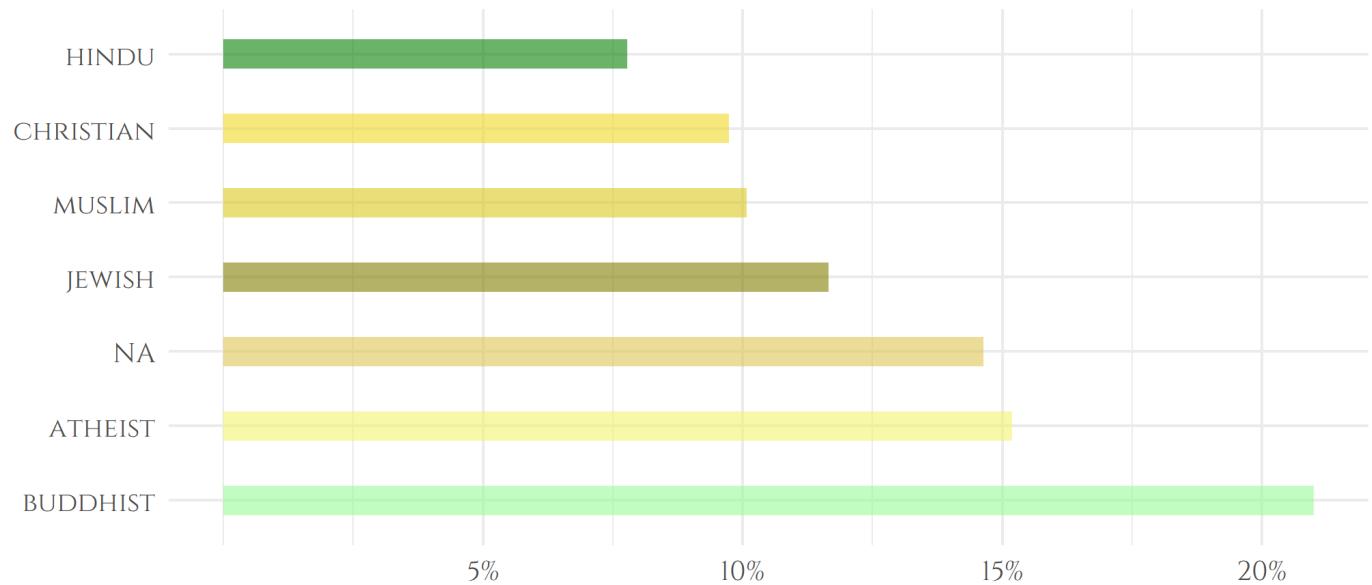
```
library(tvthemes)
# import_simpsons()
extrafont::loadfonts(device="win")

p + theme_simpsons(title.font = "Akbar",
                     text.font = "Akbar",
                     axis.text.size = 12)
```



```
ggplot(okcupid_lgbt_relig, aes(religion, pct_lgbt)) +  
  geom_bar(aes(fill = pct_lgbt), stat="identity", alpha=.6, width=1) +  
  labs(x = "", y = "", title = "This was supposed to be GoT...") +  
  scale_y_continuous(breaks = seq(0.05, 0.2, 0.05), labels=scales::percent) +  
  coord_flip() +  
  scale_fill_westeros(palette = "Tyrell", type = "continuous") +  
  theme(text = element_text(family = "Cinzel", size = 12),  
        title = element_text(family = "Cinzel", size = 14)) +  
  guides(fill=guide_colorbar("Tyrell"))
```

## THIS WAS SUPPOSED TO BE GOT...



# Common Plots à la GG

APPLICATIONS



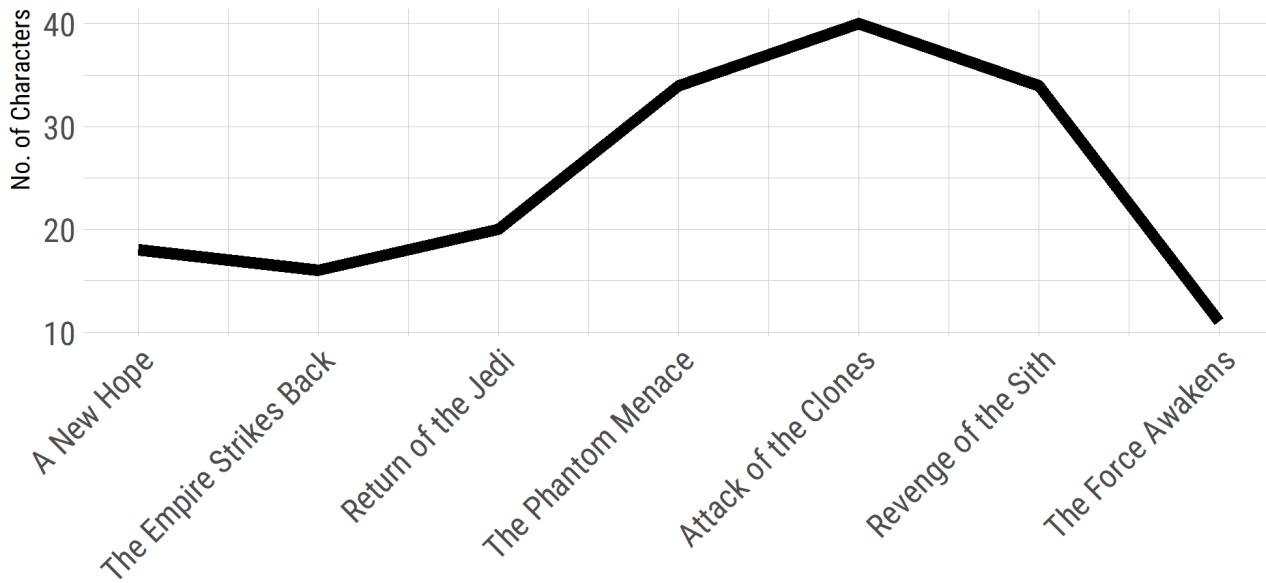
OF DATA SCIENCE

# Line Plot

```
library(hrbrthemes)

characters %>%
  group_by(film_id) %>%
  summarise(n_char = n_distinct(character_id)) %>%
  ggplot(aes(film_id, n_char)) +
  geom_line(lwd = 2) +
  labs(x = "", y = "No. of Characters",
       title = "Star Wars Films No. of Unique Characters") +
  scale_x_continuous(breaks = 1:7, labels = films$title) +
  theme_ipsum(base_family = "Roboto Condensed") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```

## Star Wars Films No. of Unique Characters



# Histogram

```
library(scales)

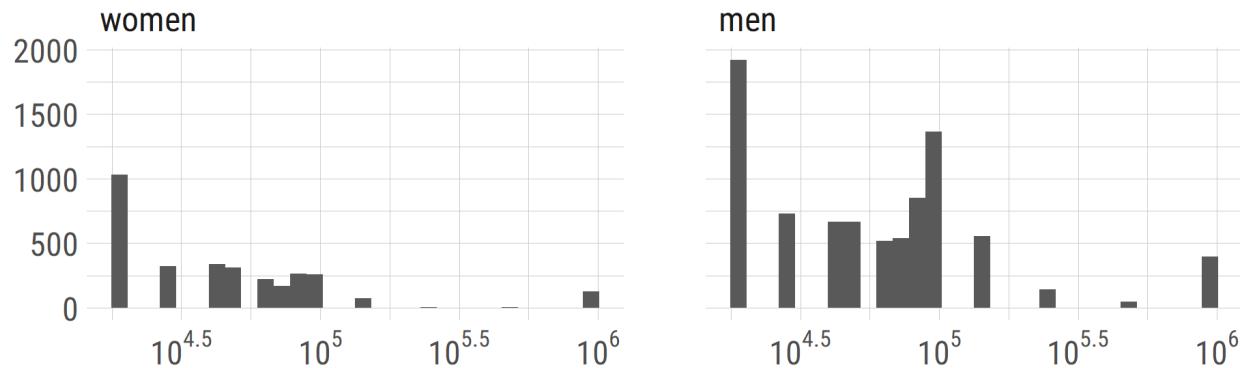
sex_names <- list(
  "m" = "men",
  "f" = "women"
)

sex_labeller <- function(variable, value){
  return(sex_names[value])
}

p2 <- okcupid %>%
  ggplot(aes(income)) +
  facet_wrap(. ~ sex, labeller = sex_labeller) +
  scale_x_log10(breaks = trans_breaks("log10", function(x) 10^x),
                 labels = trans_format("log10", math_format(10^.x)))
  labs(x = "", y = "", title = "OkCupid: Annual Income Dist. by Ge
      caption = "Income is presented as log10(Income)") +
  theme_ipsum(base_family = "Roboto Condensed")

p2 + geom_histogram()
```

## OkCupid: Annual Income Dist. by Gender

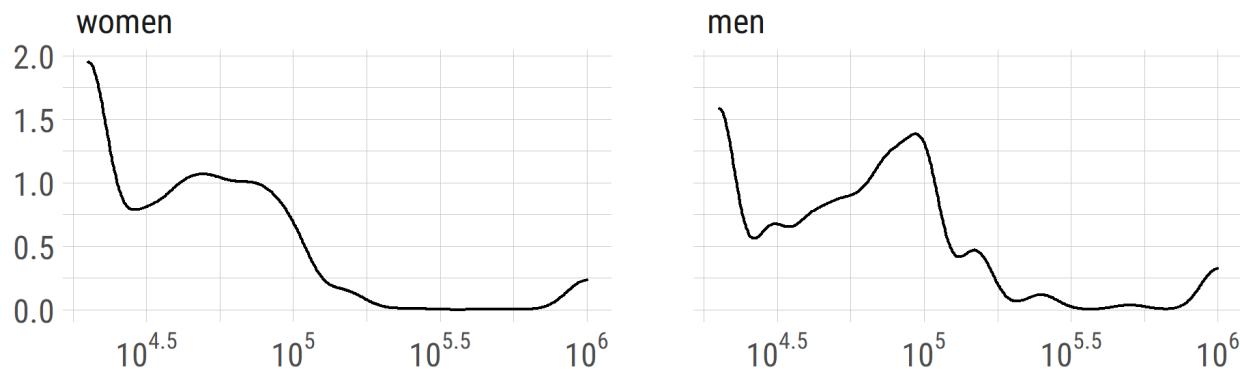


*Income is presented as  $\log_{10}(\text{Income})$*

# Density Plot

```
p2 + geom_density()
```

## OkCupid: Annual Income Dist. by Gender



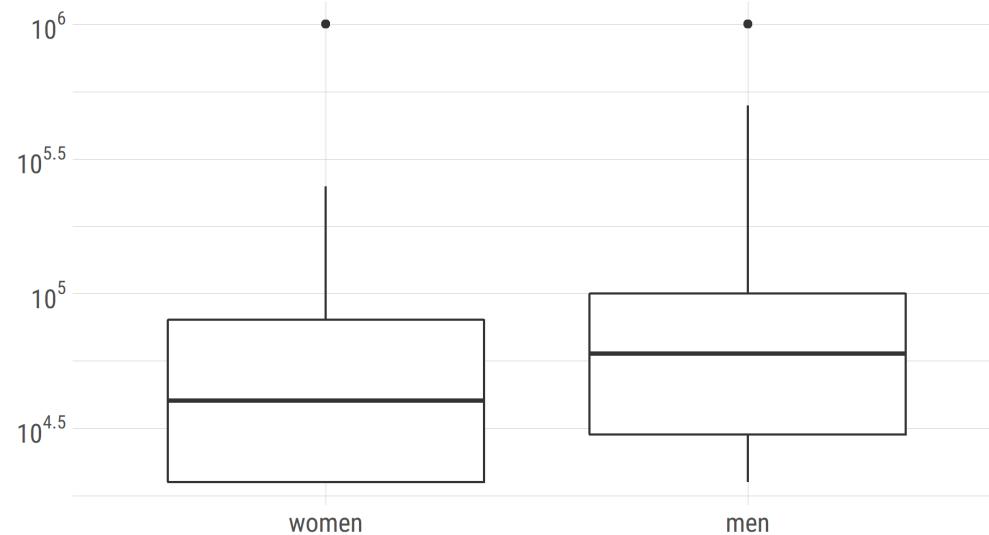
*Income is presented as  $\log_{10}(\text{Income})$*

# Box Plot

```
p3 <- okcupid %>%
  slice_sample(n = 5000) %>%
  ggplot(aes(sex, income)) +
  scale_y_log10(breaks = trans_breaks("log10", function(x) 10^x),
                 labels = trans_format("log10", math_format(10^.x)))
  labs(x = "", y = "", title = "OkCupid: Annual Income Dist. by Ge",
       caption = "Income is presented as log10(Income)") +
  scale_x_discrete(labels = c("women", "men")) +
  theme_ipsum(base_family = "Roboto Condensed")

p3 + geom_boxplot()
```

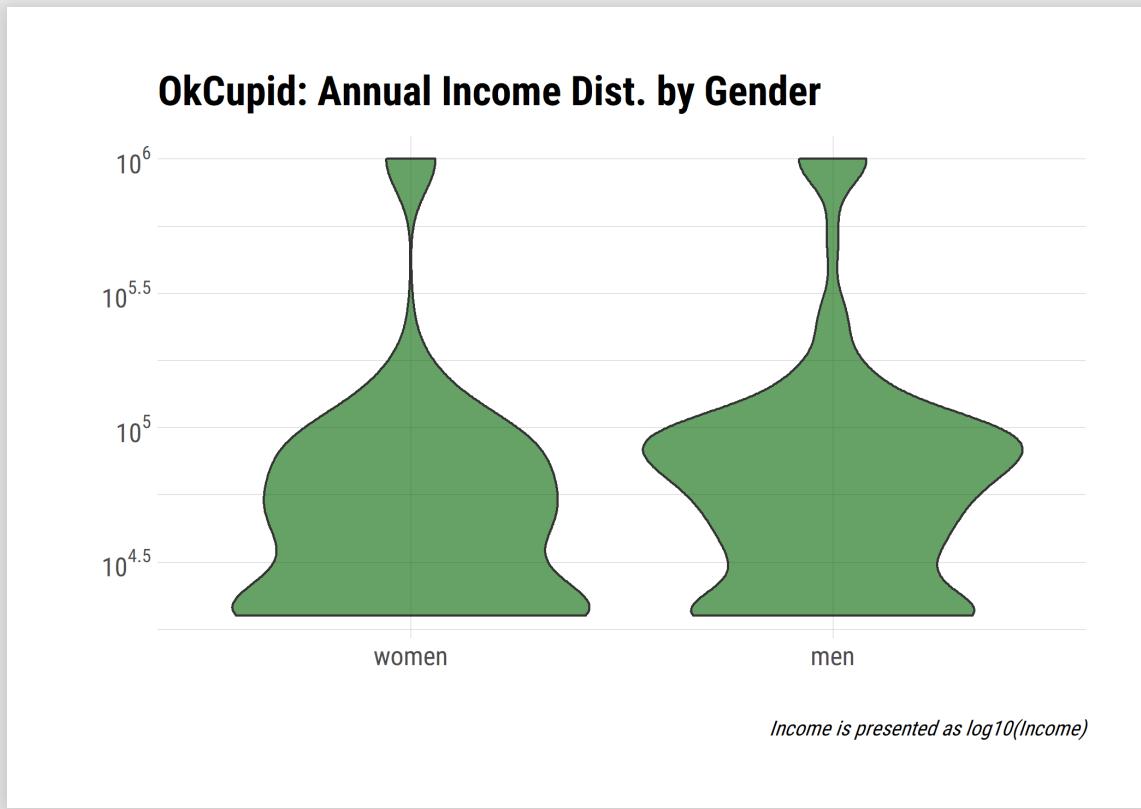
## OkCupid: Annual Income Dist. by Gender



*Income is presented as  $\log_{10}(\text{Income})$*

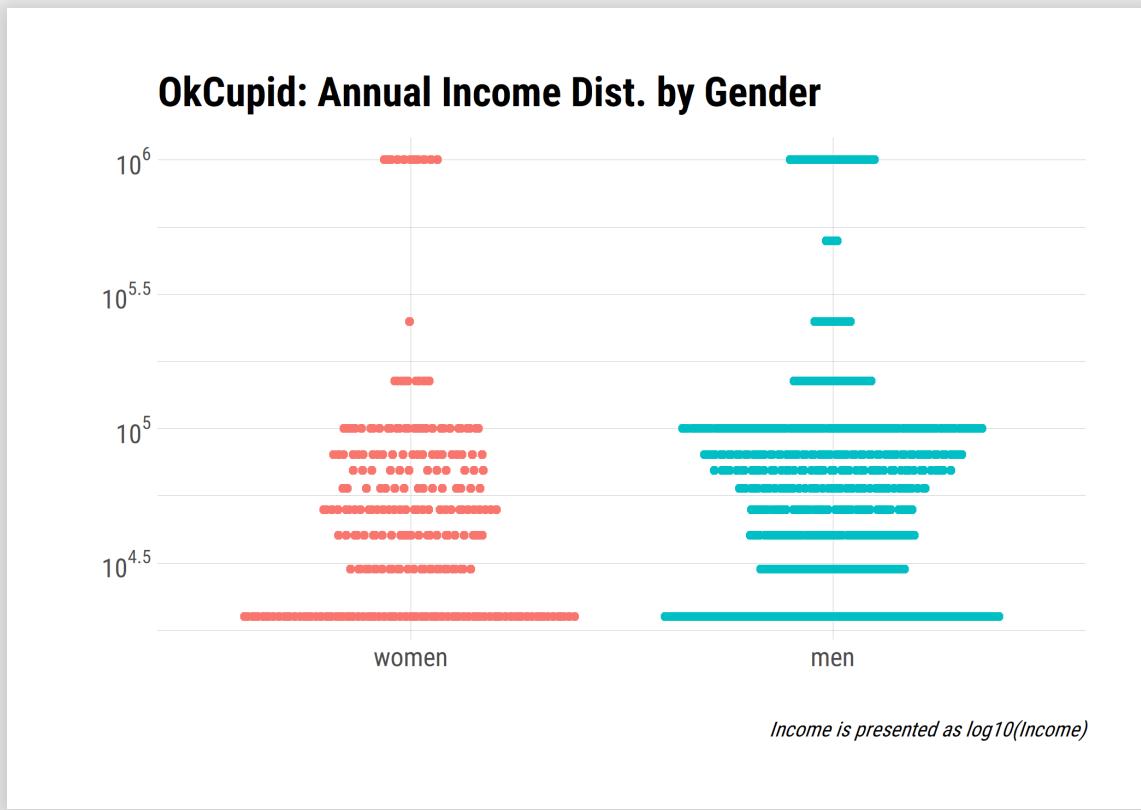
# Violin Plot

```
p3 + geom_violin(fill = "darkgreen", alpha = 0.6)
```



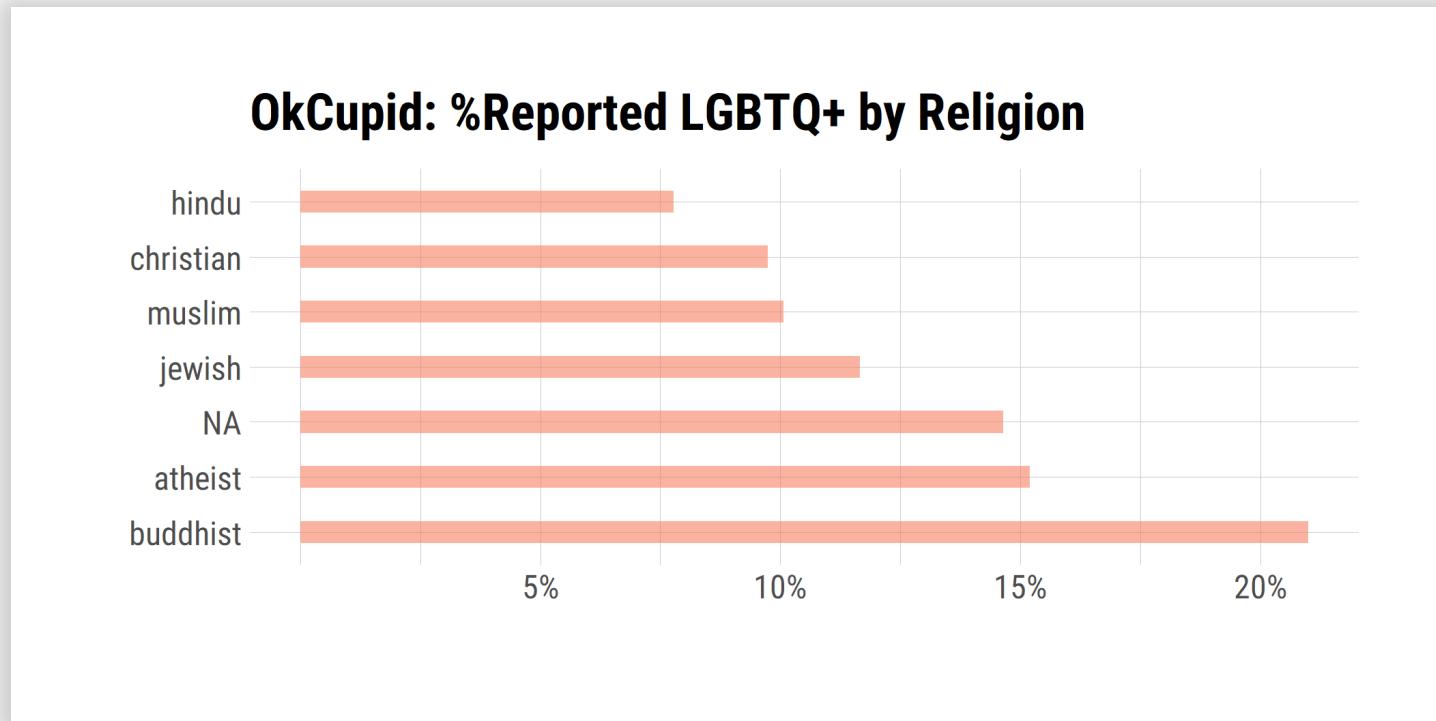
# Swarm Plot

```
p3 + ggbeeswarm::geom_quasirandom(aes(color = sex)) + guides(color
```



# Bar Plot

```
p + theme_ipsum(base_family = "Roboto Condensed")
```

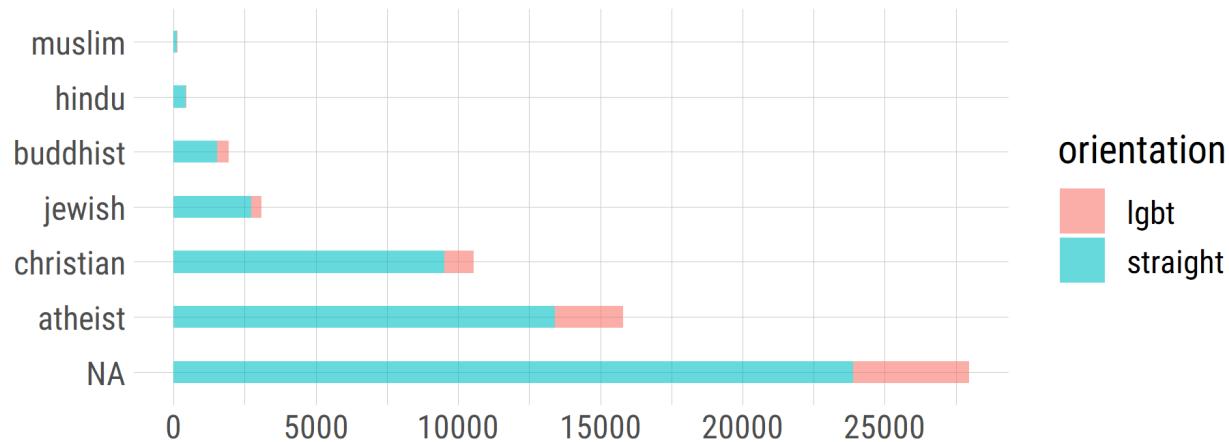


# Segmented Bar Plots: Counts

```
p4 <- ggplot(
  okcupid %>%
    mutate(orientation = recode(orientation, "gay" = "lgbt", "bisexual" = "bisexual", "asexual" = "asexual", "pansexual" = "pansexual", "omnisexual" = "omnisexual", "heterosexual" = "heterosexual"),
          religion = fct_relevel(religion2,
            levels = c("atheist", "christian", "judaean", "buddhist", "hindu", "islam", "sikh", "sufi", "yoga", "new age", "spiritual", "other", "not religious", "atheist", "christian", "judaean", "buddhist", "hindu", "islam", "sikh", "sufi", "yoga", "new age", "spiritual", "other", "not religious")),
          count(okcupid, religion2, sort = TRUE)),
    aes(religion, fill = orientation))
) +
  labs(x = "", y = "",
        title = "OkCupid: Orientation by Religion") +
  scale_y_continuous(breaks = seq(0, 25000, 5000)) +
  coord_flip() +
  theme_ipsum(base_family = "Roboto Condensed") + theme(text = element_text(family = "serif"))
```

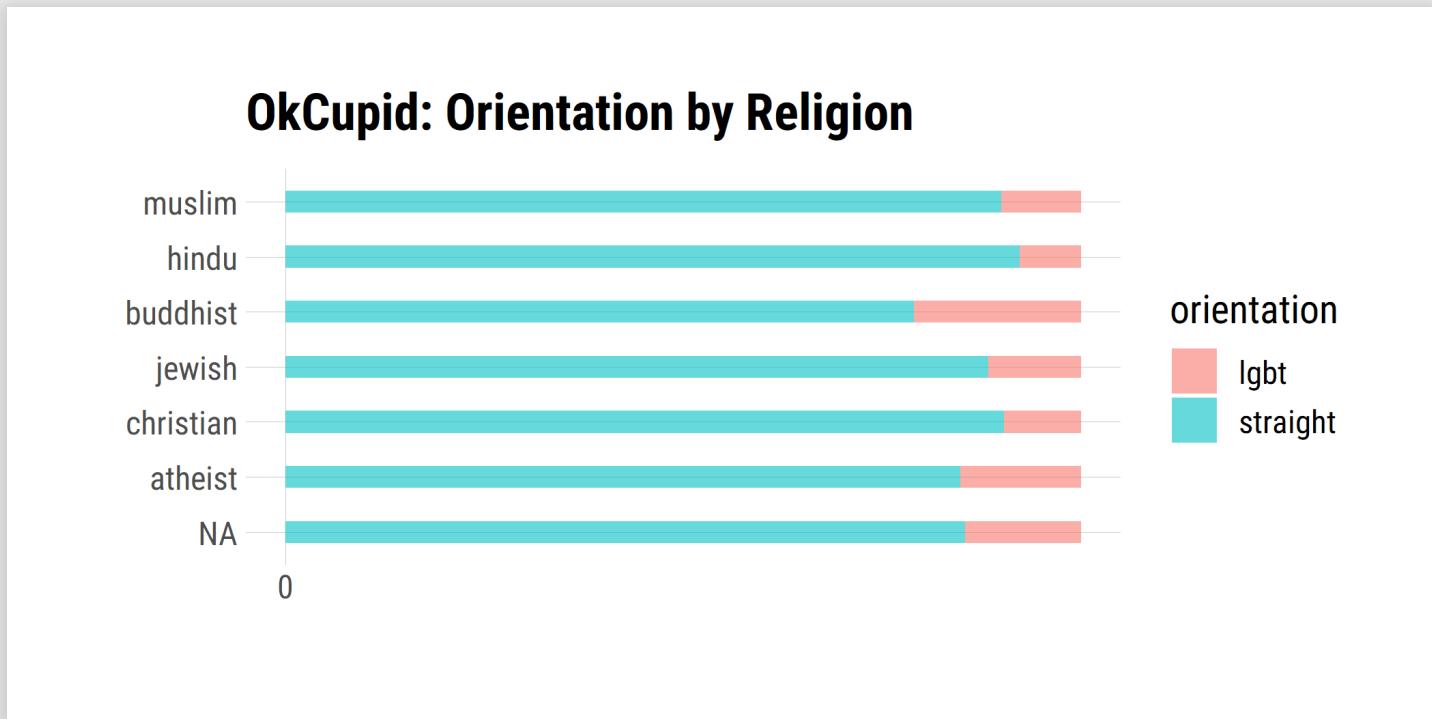
```
p4 + geom_bar(alpha=.6, width=.4)
```

## OkCupid: Orientation by Religion



# Segmented Bar Plots: Proportions

```
p4 + geom_bar(position = "fill", alpha=.6, width=.4)
```



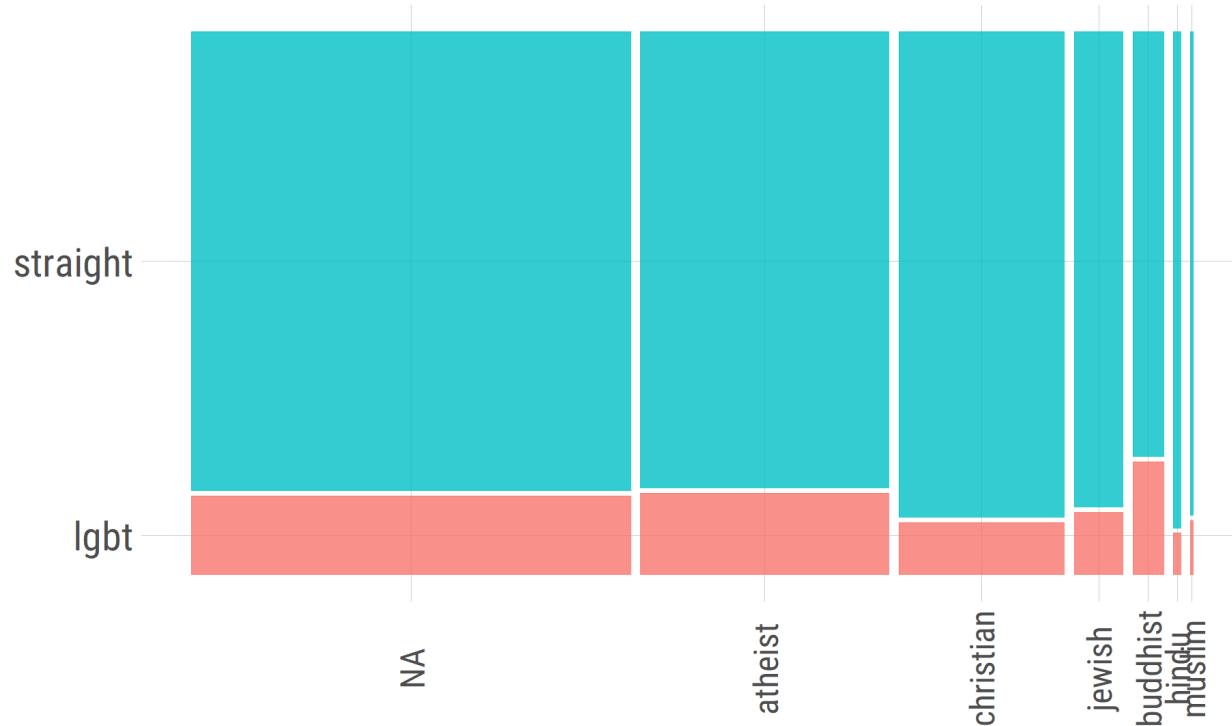
Hoping you can see why this is worse than our original plot.

# Mosaic Plots

```
library(ggmosaic)

okcupid %>%
  mutate(orientation = recode(orientation, "gay" = "lgbt", "bisexu
    religion = fct_relevel(religion2, count(okcupid, religior
ggplot() +
  geom_mosaic(aes(x = product(orientation, religion), fill=orienta
  labs(x = "", y = "", title = "OkCupid: Orientation by Religion")
  guides(fill=FALSE) +
  theme_ipsum(base_family = "Roboto Condensed") +
  theme(axis.text.y = element_text(size = 14),
        axis.text.x = element_text(angle = 90, vjust = 0.5, size =
```

## OkCupid: Orientation by Religion



# Radar Chart

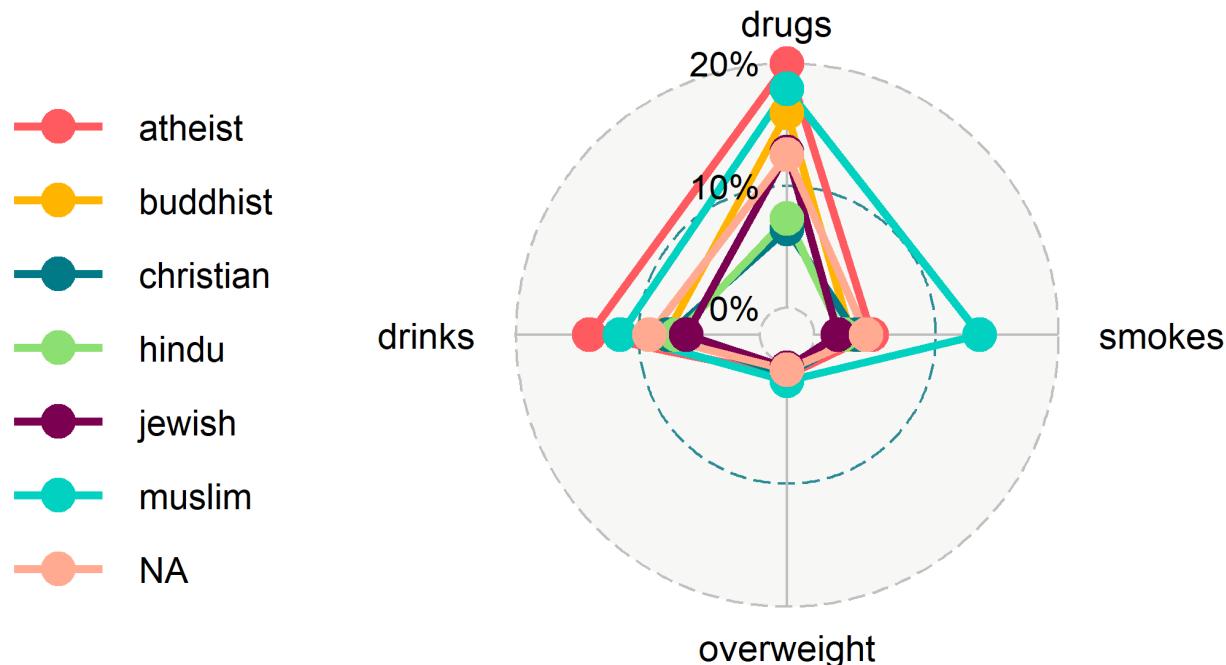
```
okcupid %>%
  group_by(religion2) %>%
  summarise(drugs = mean(drugs %in% c("sometimes", "often"), na.rm = TRUE),
            smokes = mean(smokes == "yes", na.rm = TRUE),
            overweight = mean(body_type == "overweight", na.rm = TRUE),
            drinks = mean(drinks %in% c("often", "very often", "de",
                                         "rately", "never"), na.rm = TRUE))
  ) -> okcupid_radar
okcupid_radar

## # A tibble: 7 x 5
##   religion2  drugs smokes overweight  drinks
##   <chr>      <dbl>  <dbl>     <dbl>    <dbl>
## 1 atheist     0.201  0.0473    0.0110   0.141
## 2 buddhist    0.160  0.0302    0.00494  0.0775
## 3 christian   0.0649  0.0352    0.00994  0.0759
## 4 hindu       0.0733  0.0236    0.00725  0.0689
## 5 jewish      0.128   0.0197    0.00467  0.0607
## 6 muslim      0.180   0.136     0.0155   0.115
## 7 NA          0.126   0.0427    0.00634  0.0911
```

```
library(ggradar)

ggradar(okcupid_radar, grid.max = 0.201, grid.mid = 0.10, values.
    plot.title = "OkCupid: Health Indicators by Religion") + t
```

## OkCupid: Health Indicators by Religion



# Sankey Diagram (Alluvial Plot)

```
library(ggalluvial)

g7 <- c("canada", "france", "germany", "italy", "japan", "united_kingdom"

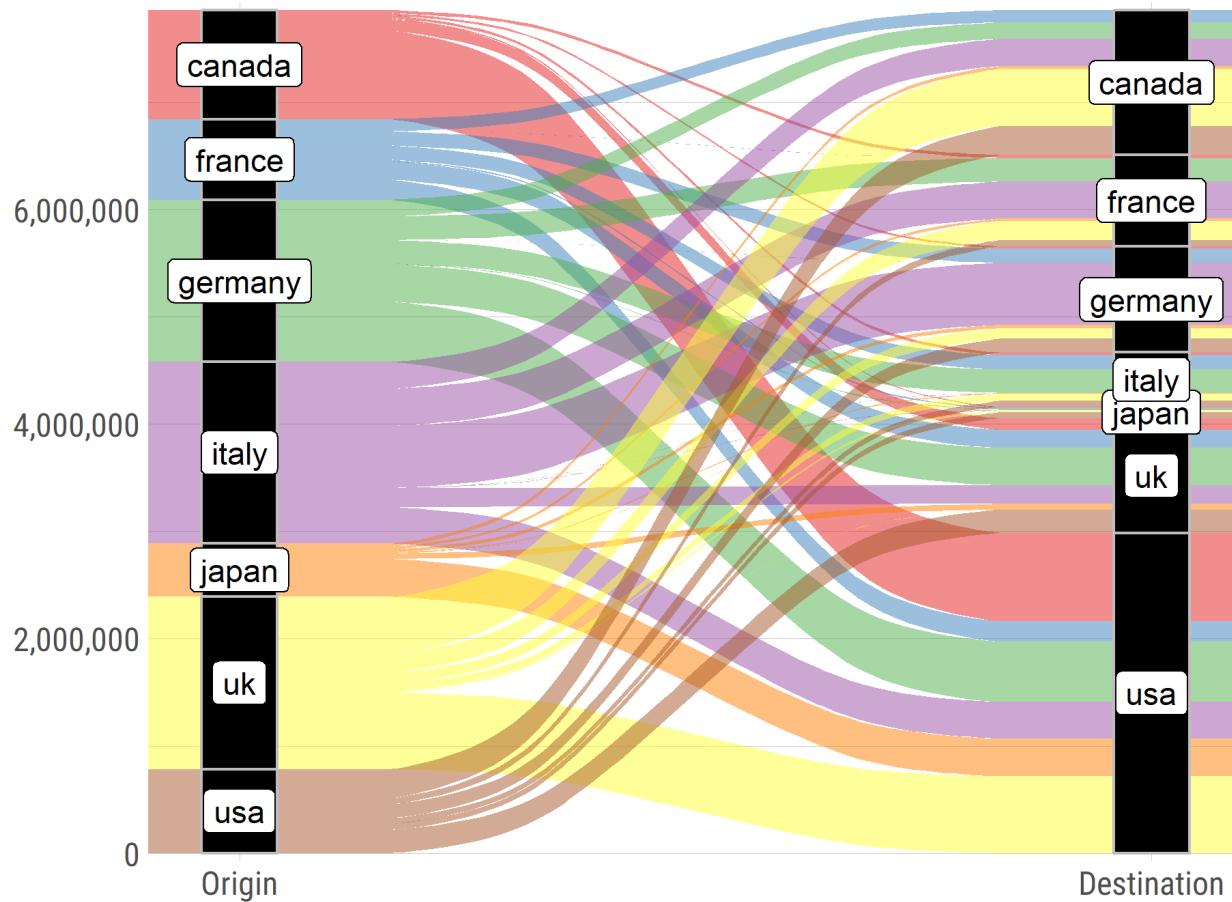
migration_2019 <- read_rds("../data/migration.rds") %>%
  pivot_longer(cols = -c(1:4), names_to = "country_orig", values_to =
  filter(year == 2019) %>%
  filter_at(c("country_orig", "country_dest"), ~.x %in% g7) %>%
  mutate_at(c("country_orig", "country_dest"), ~case_when(.x == "united_kingdom" ~ "UK", TRUE ~ .x)) %>%
  group_by(country_orig, country_dest) %>%
  summarise(n_migrants = sum(n_migrants))

migration_2019 %>% head(4)
```

```
## # A tibble: 4 x 3
## # Groups:   country_orig [1]
##   country_orig country_dest n_migrants
##   <chr>        <chr>          <dbl>
## 1 canada       canada         0
## 2 canada       france        28339
## 3 canada       germany       18211
## 4 canada       italy         25716
```

```
ggplot(migration_2019,  
       aes(y = n_migrants, axis1 = country_orig, axis2 = country_des)  
       geom_alluvium(aes(fill = country_orig)) +  
       geom_stratum(width = 1/12, fill = "black", color = "grey") +  
       geom_label(stat = "stratum", infer.label = TRUE) +  
       scale_x_discrete(limits = c("Origin", "Destination"), expand = c(0, 0)) +  
       scale_fill_brewer(type = "qual", palette = "Set1") +  
       guides(fill = FALSE) +  
       labs(y = "", title = "G7 Cross-Country Migration 2019") +  
       scale_y_comma() +  
       theme_ipsum(base_family = "Roboto Condensed")
```

## G7 Cross-Country Migration 2019



# Heatmap

```
gender_age_height_drink <- okcupid %>%
  mutate(age_group = case_when(
    age < 30 ~ "19-29",
    age < 40 ~ "30-39",
    age < 50 ~ "40-49",
    age < 60 ~ "50-59",
    TRUE ~ "60+"),
  height_group = case_when(
    height_cm < 150 ~ "149-",
    height_cm < 160 ~ "150-159",
    height_cm < 170 ~ "160-169",
    height_cm < 180 ~ "170-179",
    height_cm < 190 ~ "180-189",
    TRUE ~ "190+")) %>%
  group_by(sex, age_group, height_group) %>%
  summarise(n = n(),
            drinks = mean(drinks %in% c("often", "very often", "de
```

```
gender_age_height_drink
```

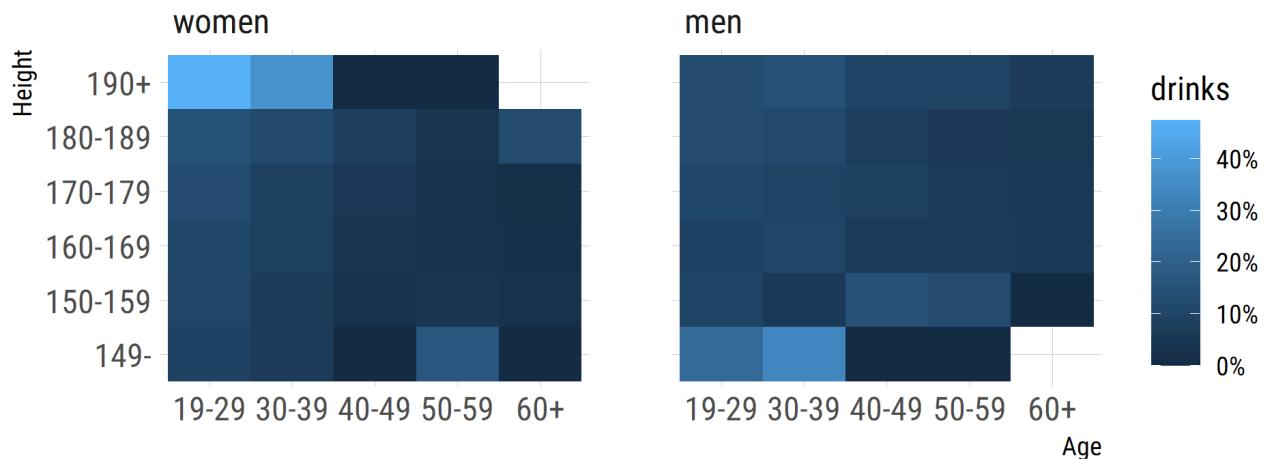
```
## # A tibble: 58 x 5
## # Groups:   sex, age_group [10]
##       sex   age_group height_group      n  drinks
##       <chr>  <chr>        <chr>     <int>    <dbl>
## 1 f       19-29       149-           163  0.0920
## 2 f       19-29       150-159       1963 0.102
## 3 f       19-29       160-169       5620 0.110
## 4 f       19-29       170-179       3113 0.130
## 5 f       19-29       180-189       310  0.145
## 6 f       19-29       190+            19  0.474
## 7 f       30-39       149-           80  0.0625
## 8 f       30-39       150-159       1293 0.0673
## 9 f       30-39       160-169       4109 0.0810
## 10 f      30-39       170-179       2231 0.0852
## # ... with 48 more rows
```

```

heat <- ggplot(gender_age_height_drink, aes(age_group, height_group))
  geom_tile(aes(text = str_c(percent(drinks, 1), " of ", n, " people")))
  facet_wrap(. ~ sex, labeller = sex_labeller) +
  labs(x = "Age", y = "Height", title = "OkCupid: %Reported Drinking by Age and Height")
  scale_fill_gradient(labels = percent_format(accuracy = 1)) +
  theme_ipsum(base_family = "Roboto Condensed")
heat

```

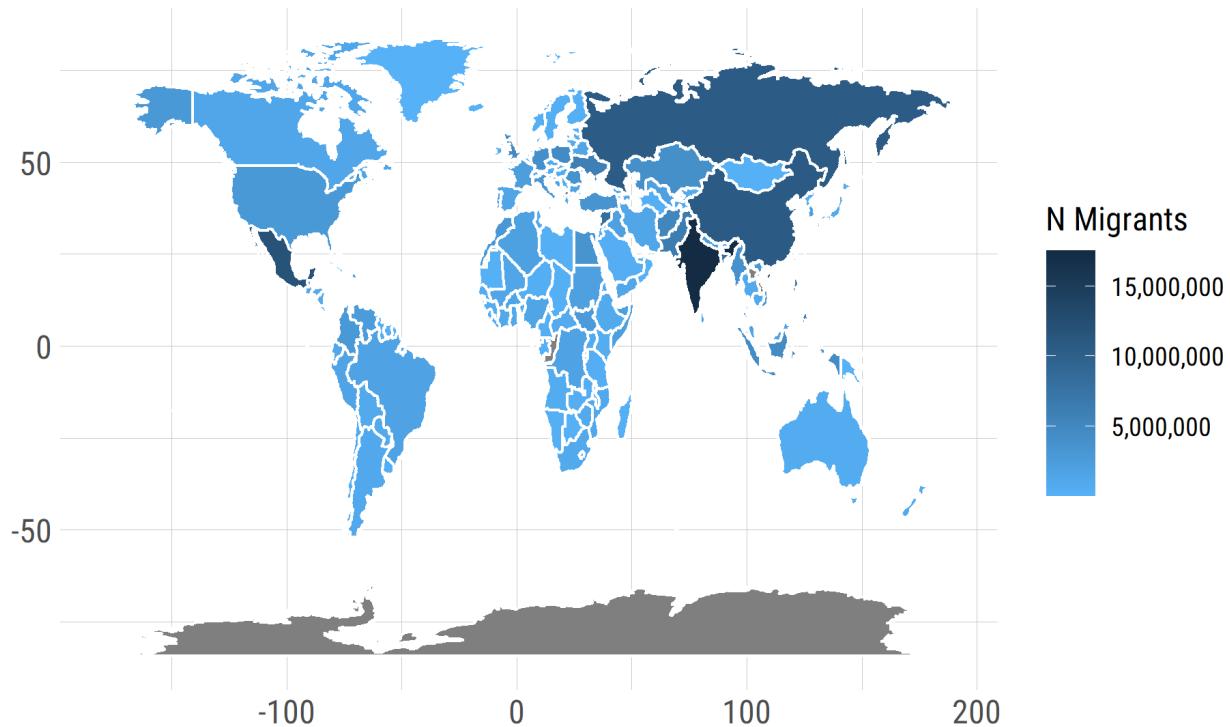
## OkCupid: %Reported Drinking by Age and Height



# An Actual Heat Map

```
clean_country_name <- function(name) {  
  str_replace_all(str_replace_all(str_to_lower(name), " |-", "_"),  
}  
  
world <- as_tibble(map_data("world"))  
  
world <- world %>%  
  mutate(region = clean_country_name(region))  
  
ggplot(world %>%  
  left_join(migration_total, "region"),  
  aes(x = long, y = lat, group = group)) +  
  geom_polygon(aes(fill = n_migrants), colour = "white") +  
  scale_fill_gradient(labels = comma, name = "N Migrants", high =  
  labs(x = "", y = "", title = "No. of Migrants by Country in 2019")  
  theme_ipsum(base_family = "Roboto Condensed")
```

## No. of Migrants by Country in 2019



# Amazing Extensions

APPLICATIONS



OF DATA SCIENCE

# patchwork

```
library(patchwork)

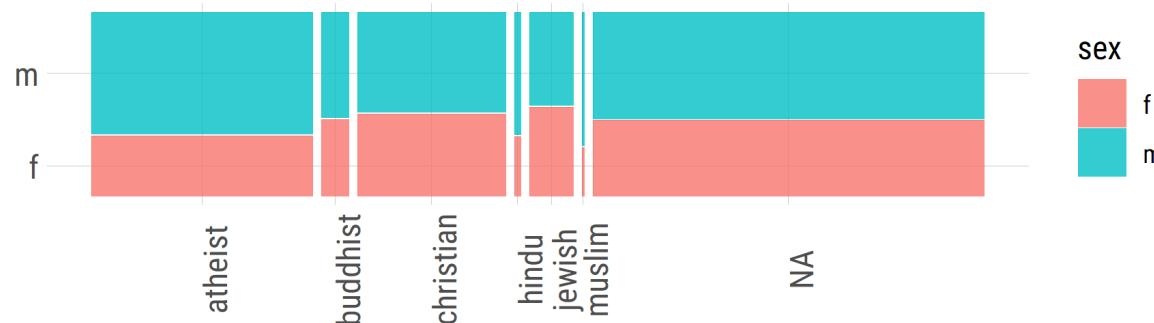
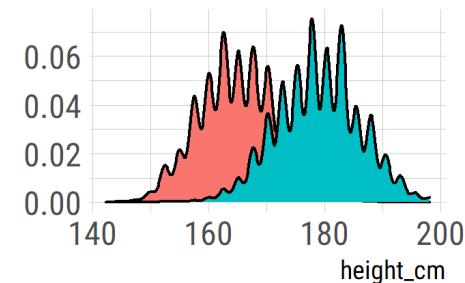
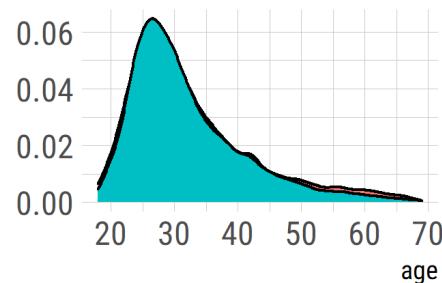
p1 <- ggplot(okcupid %>% filter(between(age, 18, 70))) +
  geom_density(aes(age, fill = sex)) +
  labs(y = "") +
  guides(fill = FALSE) +
  theme_ipsum(base_family = "Roboto Condensed")

p2 <- ggplot(okcupid %>% filter(between(height_cm, 140, 200))) +
  geom_density(aes(height_cm, fill = sex)) +
  labs(y = "") +
  guides(fill = FALSE) +
  theme_ipsum(base_family = "Roboto Condensed")

p3 <- ggplot(okcupid) +
  geom_mosaic(aes(x = product(sex, religion2), fill = sex)) +
  labs(x = "", y = "") +
  theme_ipsum(base_family = "Roboto Condensed") +
  theme(axis.text.x = element_text(angle = 90))
```

Watch this:

```
(p1 | p2) / p3
```

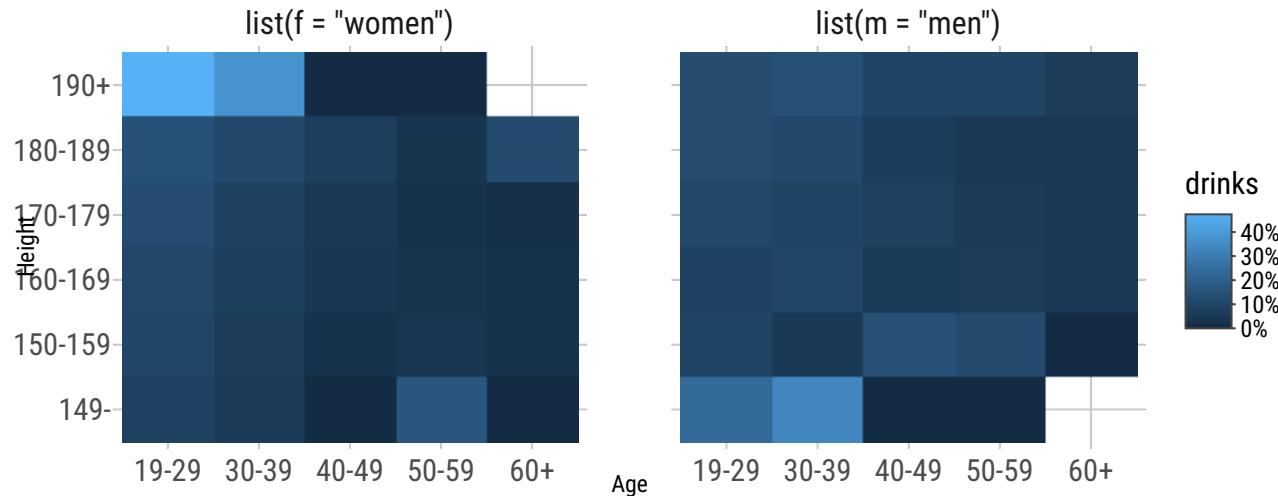


# plotly

Remember those heatmaps? They're now interactive!

```
library(plotly)
ggplotly(heat, tooltip = "text")
```

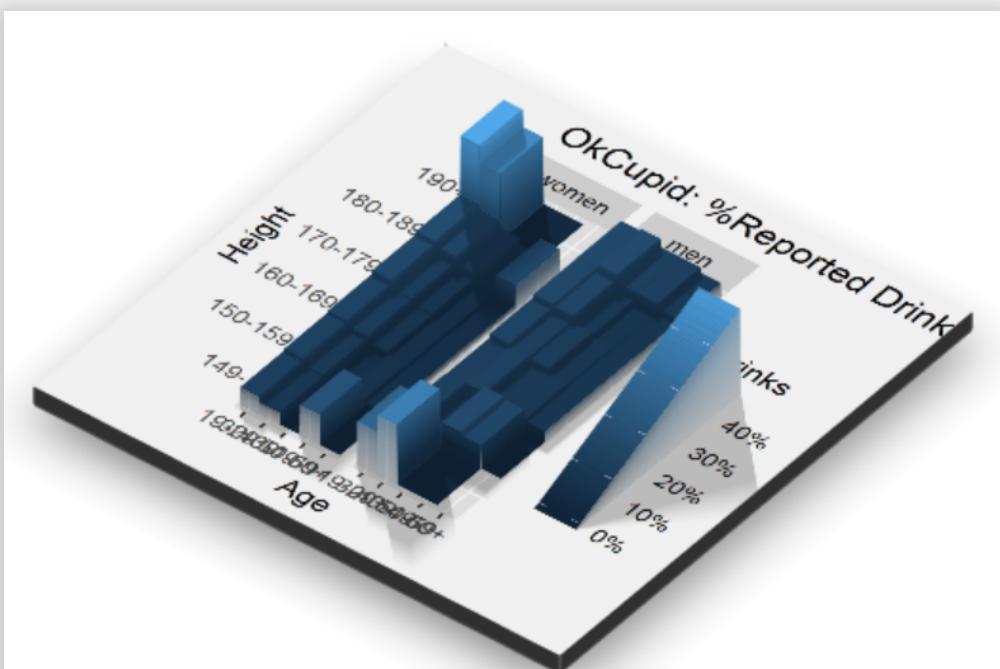
**OkCupid: %Reported Drinking by Age and Height**



# rayshader

Or rotatable 3D...

```
library(rayshader)  
plot_gg(heat)
```



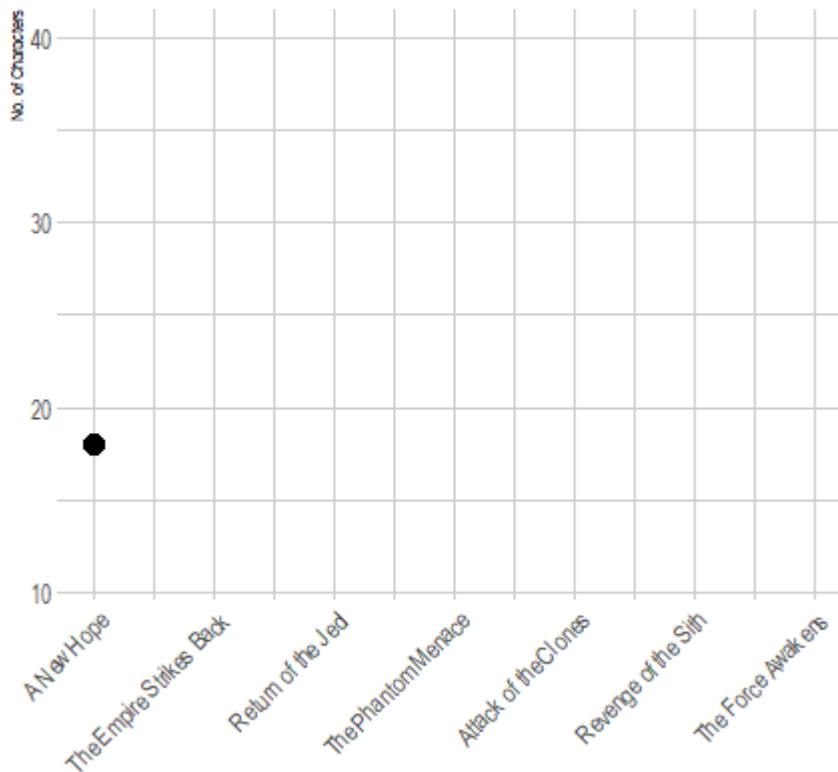
# ganimate

Take a plot and make it better:

```
library(gganimate)

characters %>%
  group_by(film_id) %>%
  summarise(n_char = n_distinct(character_id)) %>%
  ggplot(aes(film_id, n_char)) +
  geom_line(lwd = 2) +
  geom_point(aes(group = seq_along(film_id)), size = 5) +
  labs(x = "", y = "No. of Characters",
       title = "No. of Unique Characters: {films$title[frame %% 1]}",
       scale_x_continuous(breaks = 1:7, labels = films$title) +
  theme_ipsum(base_family = "Roboto Condensed") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1)) +
  transition_reveal(along = film_id) +
  ease_aes("cubic-in-out")
```

### No. of Unique Characters: A New Hope

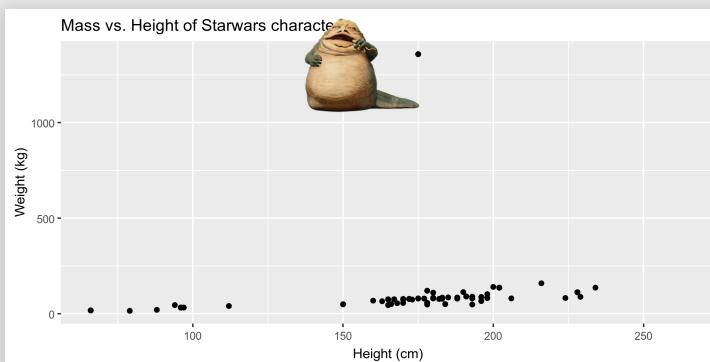


# magick

That Jabba Plot was NOT made by Photoshop...

```
library(magick)

jabba <- image_read("images/jabba.png")
fig <- image_graph(width = 2400, height = 1200, res = 300)
ggplot(data = characters, mapping = aes(x = height, y = mass)) +
  geom_point(size = 1.5) +
  labs(title = "Mass vs. Height of Starwars characters",
       x = "Height (cm)", y = "Weight (kg)")
fig %>% image_composite(jabba, offset = "+1000+30")
```



# tidygraph + ggraph

```
library(tidygraph)

characters_2_plus <- characters %>%
  add_count(name) %>%
  filter(n > 1) %>%
  select(name, film_id) %>%
  mutate(id = group_indices(., name)) # for some reason 1:n ID

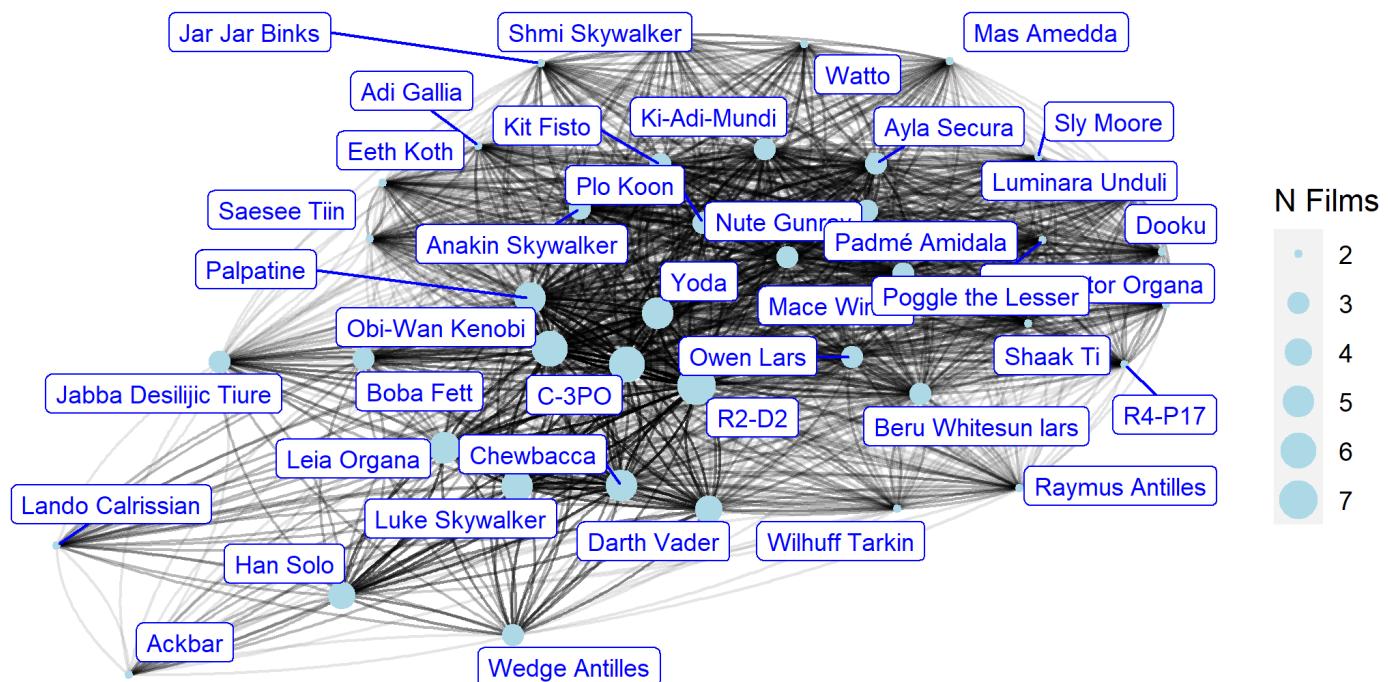
nodes <- characters_2_plus %>%
  count(id, name)

edges <- characters_2_plus %>%
  inner_join(characters_2_plus, "film_id") %>%
  rename(from = id.x,
         to = id.y) %>%
  count(from, to) %>%
  rename(weight = n) %>%
  filter(from != to)

graph <- tbl_graph(nodes = nodes, edges = edges, directed = FALSE)
```

```
library(ggraph)
ggraph(graph, layout = "fr") +
  geom_edge_fan(aes(alpha = weight), show.legend = FALSE) +
  geom_node_point(aes(size = n), color = "lightblue") +
  geom_node_label(aes(label = name), color = "blue", repel = TRUE,
  theme(panel.background = element_rect(fill="white", colour = "wl
  labs(size = "N Films", title = "Star Wars: Characters Network",
```

## Star Wars: Characters Network



An Edge between two characters means they appeared in the same film

# ggwithimages

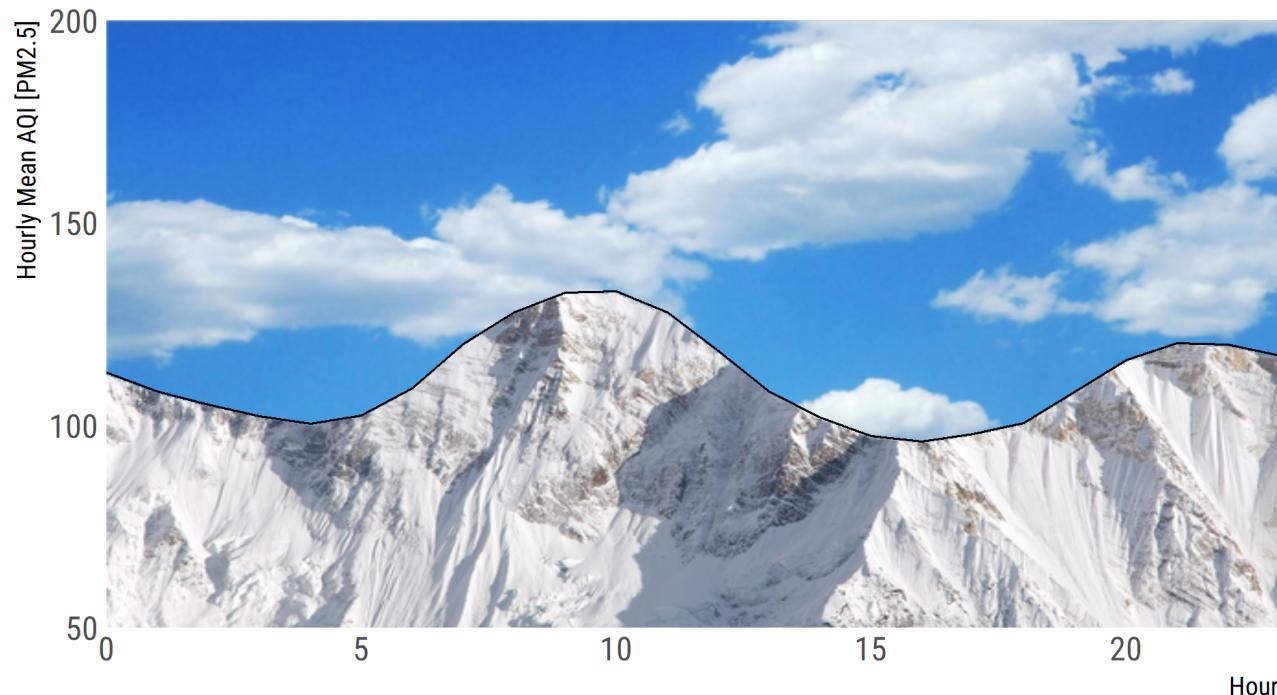
```
library(ggwithimages)

annapurna <- png:::readPNG(system.file("extdata", "annapurna.png",
sky <- png:::readPNG(system.file("extdata", "sky.png", package =
kathmandu_hourly_aqi <- readr:::read_csv(system.file("extdata", "ka

ggplot(kathmandu_hourly_aqi, aes(hour, aqi)) +
  geom_line_with_image(annapurna, sky) +
  labs(title = "Air Quality Index in the Thamel, Kathmandu, Nepal",
       subtitle = "Measured in PM2.5 by the US Embassy in Kathmandu",
       y = "Hourly Mean AQI [PM2.5]",
       x = "Hour") +
  ylim(c(50, 200)) +
  theme_ipsum(base_family = "Roboto Condensed")
```

# Air Quality Index in the Thamel, Kathmandu, Nepal

Measured in PM2.5 by the US Embassy in Kathmandu



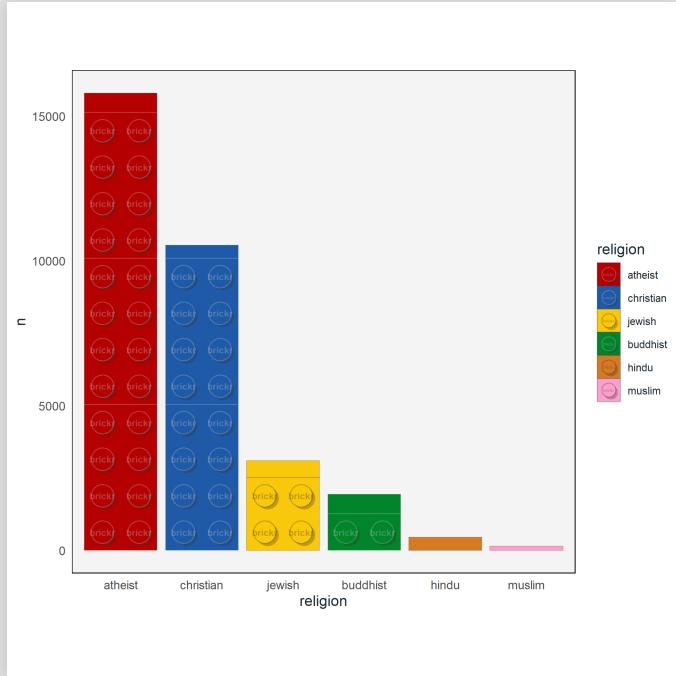
# brickr...

```
library(brickr)

okcupid %>%
  count(religion2) %>%
  filter(religion2 != "NA") %>%
  mutate(religion =
    fct_relevel(religion2, c("atheist", "christian", "jewis
ggplot(aes(religion, n)) +
  geom_brick_col(aes(fill = religion)) +
  scale_fill_brick() +
  coord_brick() +
  theme_brick()
```

(Though this was available only on v0.2.0 🎉)

And... it can get a lot more fun:



Get the legos you need with  
`build_instructions()`