# R for Data Science - Manual

## Basic information

### Useful resources

!!! The most useful book for R markdown - "R Markdown: The Definitive Guide" Ebook "R for Graduate Students"
Ebook "R for Data Science"
Ebook "Statistical Inference via Data Science: A ModernDive into R and Tidyverse"
Gradient generator
ggthemes
R studio cheatsheets
Introduction to ggplot2
Themes for Bootstrap

### Versions and plugins

To update R version: go to R GUI (old), run 1) install.packages("installr"); library(installr); choose installr option on the tab next to the option "Help", then update... voila!

### Packages installation and loading:

```
getRversion()

install.packages(c("afex", "tidyverse", "emmeans", "writexl",
    "readxl", "ggthemes", "ggplot2", "extrafont", "reshape",
    "reshape2", "gcookbook", "readr", "swirl", "ggpubr", "lubridate",
    "rmarkdown", "formatR", "knitr", "magrittr", "glue", "viridis",
    "ggsci", "gcookbook"))

install.packages("plotly")
```

```
library(afex)
library(emmeans)
library(tidyverse)
library(writexl)
library(readxl)
library(ggthemes)
library(extrafont)
library(reshape)
library(reshape2)
library(gcookbook)
```

```
library(readr)
library(swirl)
library(ggpubr)
library(ggthemes)
library(lubridate)
library(rmarkdown)
library(formatR)
library(knitr)
library(magrittr)
library(glue)
library(viridis)
library(ggsci)
library(gcookbook)
library(plotly)

# font_import() # for extrafont package loadfonts(device =
# 'win') # loading fonts from extrafont
```

**Shortcuts**

`Ctrl + Alt + I` : insert an R code chunk `Ctrl + Shift + Enter` : run the entire code chunk

# Variables

```
var1 = "string_value"
```

# Data types

Check the data type of a variable: `class(variable)`

**Character (str)**

**Methods**: `length()`

```
# STRING
var1 = "string"
class(var1)
```

```
## [1] "character"
```

```
# f-string
a = 5
b = glue("Value: {a}")
print(b)
```

```
## Value: 5
```

**Integer**

**Numeric**

**Logical**

```
a = TRUE
class(a)
```

```
## [1] "logical"
```

**Vector**

Composed of elements of the same type Slicing is 1-based, so the the starting index is 1 instead of the usual 0.

Methods: - `length(listname)`

```
listNames <- c("a", "b", "c", "d", "e", "f")
listNames[1]
```

```
## [1] "a"
```

```
listNames[1:3]  # Slice the list
```

```
## [1] "a" "b" "c"
```

```
listNames[c(1, 3, 6)]  # Choose elements at indices 1, 3, and 6
```

```
## [1] "a" "c" "f"
```

**List**

Composed of elements of different types.

```
listNames <- list("a", 15, c("1", "2", "3"), "d", "e", "f")
class(listNames)
```

```
## [1] "list"
```

```
listNames[[3]]  # To slice, usually we use double brackets
```

```
## [1] "1" "2" "3"
```

**Matrix**

```r
m1 = matrix(c(1, 2, 3, 4, 5, 6), nrow = 2, ncol = 3, byrow = TRUE)
m1
```

```
##      [,1] [,2] [,3]
## [1,]    1    2    3
## [2,]    4    5    6
```

```r
m1[1, 3]   # Row 1, column 3
```

```
## [1] 3
```

???

```r
# DATASET Create brand-new dataset
data <- data.frame(Chr = c("Chr1", "Chr2", "Chr2", "Chr3", "Chr4",
    "Chr11"), Type = c("1_Promoter", "1_Promoter", "2_Enhancer",
    "3_Activator", "1_Promoter", "3_Activator"), Value = c(1e+09,
    5e+10, 6e+09, 9e+10, 8e+10, 5e+10))

data$Type
```

```
## [1] "1_Promoter"  "1_Promoter"  "2_Enhancer"  "3_Activator" "1_Promoter"
## [6] "3_Activator"
```

# Operators

## Assignment operators

```r
# Assignment operators
a = 5
a <- 5
class(a)
```

```
## [1] "numeric"
```

## Math operators

```r
# Math operators
a = 5
b = 10
c = a + b
c
```

```
## [1] 15
```

**Relational operators** >, <, ==, !=

**Logical operators** & (and), | (or)

# Flow statements

## IF

```
a = 1

if (a > 2) {
    print("'a' is greater than 2")
} else {
    print("not")
}
```

```
## [1] "not"
```

## FOR

```
list1 = c(1, 2, 3)

for (i in list1) {
    print(i)
}
```

```
## [1] 1
## [1] 2
## [1] 3
```

## WHILE

```
i = 1

while (i < 5) {
    print(i)
    i = i + 1
}
```

```
## [1] 1
## [1] 2
## [1] 3
## [1] 4
```

# Modules and packages

**Pre-loaded datasets:**

```
data()  # check a list of pre-loaded datasets
```

## Dplyr ( = pandas)

This is basically the analogue to Python's Pandas.

```
library(dplyr)

# Read a local file df <- read.csv('Data/diamonds.csv')
# Load a pre-loaded dataset from a package
starwars  # from the dplyr package
```

```
## # A tibble: 87 x 14
##    name          height  mass hair_~1 skin_~2 eye_c~3 birth~4 sex    gender homew~5
##    <chr>          <int> <dbl> <chr>   <chr>   <chr>     <dbl> <chr>  <chr>  <chr>
##  1 Luke Skywa~      172    77 blond   fair    blue         19 male   mascu~ Tatooi~
##  2 C-3PO            167    75 <NA>    gold    yellow      112 none   mascu~ Tatooi~
##  3 R2-D2             96    32 <NA>    white,~ red          33 none   mascu~ Naboo
##  4 Darth Vader      202   136 none    white   yellow     41.9 male   mascu~ Tatooi~
##  5 Leia Organa      150    49 brown   light   brown        19 fema~  femin~ Aldera~
##  6 Owen Lars        178   120 brown,~ light   blue         52 male   mascu~ Tatooi~
##  7 Beru White~      165    75 brown   light   blue         47 fema~  femin~ Tatooi~
##  8 R5-D4             97    32 <NA>    white,~ red          NA none   mascu~ Tatooi~
##  9 Biggs Dark~      183    84 black   light   brown        24 male   mascu~ Tatooi~
## 10 Obi-Wan Ke~      182    77 auburn~ fair    blue-g~      57 male   mascu~ Stewjon
## # ... with 77 more rows, 4 more variables: species <chr>, films <list>,
## #   vehicles <list>, starships <list>, and abbreviated variable names
## #   1: hair_color, 2: skin_color, 3: eye_color, 4: birth_year, 5: homeworld
```

```
dim(df)  # Print dimensions of the dataset
```

```
## NULL
```

```
head(df)  # Print the first 5 rows
```

```
##
## 1 function (x, df1, df2, ncp, log = FALSE)
## 2 {
## 3     if (missing(ncp))
## 4         .Call(C_df, x, df1, df2, log)
## 5     else .Call(C_dnf, x, df1, df2, ncp, log)
## 6 }
```

```
# Filter
filter(starwars, height > 170, mass > 100)
```

```
## # A tibble: 10 x 14
##    name          height  mass hair_~1 skin_~2 eye_c~3 birth~4 sex    gender homew~5
##    <chr>          <int> <dbl> <chr>   <chr>   <chr>     <dbl> <chr>  <chr>  <chr>
```

```
##  1 Darth Vader     202  136 none    white   yellow    41.9 male  mascu~ Tatooi~
##  2 Owen Lars       178  120 brown,~ light   blue      52   male  mascu~ Tatooi~
##  3 Chewbacca       228  112 brown   unknown blue     200   male  mascu~ Kashyy~
##  4 Jabba Desi~     175 1358 <NA>    green-~ orange   600   herm~ mascu~ Nal Hu~
##  5 Jek Tono P~     180  110 brown   fair    blue      NA   male  mascu~ Bestin~
##  6 IG-88           200  140 none    metal   red       15   none  mascu~ <NA>
##  7 Bossk           190  113 none    green   red       53   male  mascu~ Trando~
##  8 Dexter Jet~     198  102 none    brown   yellow    NA   male  mascu~ Ojom
##  9 Grievous        216  159 none    brown,~ green,~   NA   male  mascu~ Kalee
## 10 Tarfful         234  136 brown   brown   blue      NA   male  mascu~ Kashyy~
## # ... with 4 more variables: species <chr>, films <list>, vehicles <list>,
## #   starships <list>, and abbreviated variable names 1: hair_color,
## #   2: skin_color, 3: eye_color, 4: birth_year, 5: homeworld
```

```
filter(starwars, hair_color == "black" | hair_color == "brown")
```

```
## # A tibble: 31 x 14
##    name        height  mass hair_~1 skin_~2 eye_c~3 birth~4 sex   gender homew~5
##    <chr>        <int> <dbl> <chr>   <chr>   <chr>     <dbl> <chr> <chr>  <chr>
##  1 Leia Organa    150    49 brown   light   brown        19 fema~ femin~ Aldera~
##  2 Beru White~    165    75 brown   light   blue         47 fema~ femin~ Tatooi~
##  3 Biggs Dark~    183    84 black   light   brown        24 male  mascu~ Tatooi~
##  4 Chewbacca      228   112 brown   unknown blue        200 male  mascu~ Kashyy~
##  5 Han Solo       180    80 brown   fair    brown        29 male  mascu~ Corell~
##  6 Wedge Anti~    170    77 brown   fair    hazel        21 male  mascu~ Corell~
##  7 Jek Tono P~    180   110 brown   fair    blue         NA male  mascu~ Bestin~
##  8 Boba Fett      183  78.2 black   fair    brown      31.5 male  mascu~ Kamino
##  9 Lando Calr~    177    79 black   dark    brown        31 male  mascu~ Socorro
## 10 Arvel Cryn~     NA    NA brown   fair    brown        NA male  mascu~ <NA>
## # ... with 21 more rows, 4 more variables: species <chr>, films <list>,
## #   vehicles <list>, starships <list>, and abbreviated variable names
## #   1: hair_color, 2: skin_color, 3: eye_color, 4: birth_year, 5: homeworld
```

```
# Select columns
select(starwars, height, mass)
```

```
## # A tibble: 87 x 2
##    height  mass
##     <int> <dbl>
##  1    172    77
##  2    167    75
##  3     96    32
##  4    202   136
##  5    150    49
##  6    178   120
##  7    165    75
##  8     97    32
##  9    183    84
## 10    182    77
## # ... with 77 more rows
```

```
### Select columns that start or end with ...
select(starwars, starts_with("s"))
```

```
## # A tibble: 87 x 4
##    skin_color   sex    species starships
##    <chr>        <chr>  <chr>   <list>
##  1 fair         male   Human   <chr [2]>
##  2 gold         none   Droid   <chr [0]>
##  3 white, blue  none   Droid   <chr [0]>
##  4 white        male   Human   <chr [1]>
##  5 light        female Human   <chr [0]>
##  6 light        male   Human   <chr [0]>
##  7 light        female Human   <chr [0]>
##  8 white, red   none   Droid   <chr [0]>
##  9 light        male   Human   <chr [1]>
## 10 fair         male   Human   <chr [5]>
## # ... with 77 more rows
```

```
select(starwars, ends_with("X"))
```

```
## # A tibble: 87 x 1
##    sex
##    <chr>
##  1 male
##  2 none
##  3 none
##  4 male
##  5 female
##  6 male
##  7 female
##  8 none
##  9 male
## 10 male
## # ... with 77 more rows
```

```
# Choose a randomly-selected sample
sample_n(starwars, 5)  # A numbered sample
```

```
## # A tibble: 5 x 14
##   name          height  mass hair_~1 skin_~2 eye_c~3 birth~4 sex    gender homew~5
##   <chr>          <int> <dbl> <chr>   <chr>   <chr>     <dbl> <chr>  <chr>  <chr>
## 1 Mace Windu       188    84 none    dark    brown        72 male   mascu~ Haruun~
## 2 Greedo           173    74 <NA>    green   black        44 male   mascu~ Rodia
## 3 Beru Whites~     165    75 brown   light   blue         47 fema~  femin~ Tatooi~
## 4 Wilhuff Tar~     180    NA auburn~ fair    blue         64 male   mascu~ Eriadu
## 5 Dormé            165    NA brown   light   brown        NA fema~  femin~ Naboo
## # ... with 4 more variables: species <chr>, films <list>, vehicles <list>,
## #   starships <list>, and abbreviated variable names 1: hair_color,
## #   2: skin_color, 3: eye_color, 4: birth_year, 5: homeworld
```

```r
sample_frac(starwars, 0.01)   # Fraction of the dataset
```

```
## # A tibble: 1 x 14
##   name   height  mass hair_color skin_color eye_co~1 birth~2 sex    gender homew~3
##   <chr>  <int> <dbl> <chr>      <chr>      <chr>      <dbl> <chr> <chr> <chr>
## 1 Cordé    157    NA brown      light      brown         NA fema~ femin~ Naboo
## # ... with 4 more variables: species <chr>, films <list>, vehicles <list>,
## #   starships <list>, and abbreviated variable names 1: eye_color,
## #   2: birth_year, 3: homeworld
```

```r
drop_na(starwars, sex)   # Deal with NAN values
```

```
## # A tibble: 83 x 14
##    name         height  mass hair_~1 skin_~2 eye_c~3 birth~4 sex   gender homew~5
##    <chr>         <int> <dbl> <chr>   <chr>   <chr>     <dbl> <chr> <chr> <chr>
##  1 Luke Skywa~     172    77 blond   fair    blue         19 male  mascu~ Tatooi~
##  2 C-3PO           167    75 <NA>    gold    yellow      112 none  mascu~ Tatooi~
##  3 R2-D2            96    32 <NA>    white,~ red          33 none  mascu~ Naboo
##  4 Darth Vader     202   136 none    white   yellow     41.9 male  mascu~ Tatooi~
##  5 Leia Organa     150    49 brown   light   brown        19 fema~ femin~ Aldera~
##  6 Owen Lars       178   120 brown,~ light   blue         52 male  mascu~ Tatooi~
##  7 Beru White~     165    75 brown   light   blue         47 fema~ femin~ Tatooi~
##  8 R5-D4            97    32 <NA>    white,~ red          NA none  mascu~ Tatooi~
##  9 Biggs Dark~     183    84 black   light   brown        24 male  mascu~ Tatooi~
## 10 Obi-Wan Ke~     182    77 auburn~ fair    blue-g~      57 male  mascu~ Stewjon
## # ... with 73 more rows, 4 more variables: species <chr>, films <list>,
## #   vehicles <list>, starships <list>, and abbreviated variable names
## #   1: hair_color, 2: skin_color, 3: eye_color, 4: birth_year, 5: homeworld
```

An example of a pipeline of a dataset processing:

```r
males <- starwars %>%
    filter(mass < 80, sex == "male") %>%
    drop_na(hair_color)
males
```

```
## # A tibble: 18 x 14
##    name         height  mass hair_~1 skin_~2 eye_c~3 birth~4 sex   gender homew~5
##    <chr>         <int> <dbl> <chr>   <chr>   <chr>     <dbl> <chr> <chr> <chr>
##  1 Luke Skywa~     172    77 blond   fair    blue         19 male  mascu~ Tatooi~
##  2 Obi-Wan Ke~     182    77 auburn~ fair    blue-g~      57 male  mascu~ Stewjon
##  3 Wedge Anti~     170    77 brown   fair    hazel        21 male  mascu~ Corell~
##  4 Yoda             66    17 white   green   brown       896 male  mascu~ <NA>
##  5 Palpatine       170    75 grey    pale    yellow       82 male  mascu~ Naboo
##  6 Boba Fett       183  78.2 black   fair    brown      31.5 male  mascu~ Kamino
##  7 Lando Calr~     177    79 black   dark    brown        31 male  mascu~ Socorro
##  8 Lobot           175    79 none    light   blue         37 male  mascu~ Bespin
##  9 Wicket Sys~      88    20 brown   brown   brown         8 male  mascu~ Endor
## 10 Nien Nunb       160    68 none    grey    black        NA male  mascu~ Sullust
## 11 Jar Jar Bi~     196    66 none    orange  orange       52 male  mascu~ Naboo
## 12 Sebulba         112    40 none    grey, ~ orange       NA male  mascu~ Malast~
## 13 Dud Bolt         94    45 none    blue, ~ yellow       NA male  mascu~ Vulpter
```

```
## 14 Ben Quadin~    163 65   none    grey, ~ orange     NA   male  mascu~ Tund
## 15 Jango Fett     183 79   black   tan     brown      66   male  mascu~ Concor~
## 16 Ratts Tyer~     79 15   none    grey, ~ unknown    NA   male  mascu~ Aleen ~
## 17 Wat Tambor     193 48   none    green,~ unknown    NA   male  mascu~ Skako
## 18 Raymus Ant~    188 79   brown   light   brown      NA   male  mascu~ Aldera~
## # ... with 4 more variables: species <chr>, films <list>, vehicles <list>,
## #   starships <list>, and abbreviated variable names 1: hair_color,
## #   2: skin_color, 3: eye_color, 4: birth_year, 5: homeworld
```

# Functionality

- Ctrl + shift + R : creates headers for your R script

- Alt + shift + K : displays all programmed keyboard shortcuts for R

- Ctrl + shift + M : write "%>%" (tidyverse pipe)

# Basic commands

- Save as .csv: `write_csv(name, "filename.csv")`
- Import data from .txt file: `read.csv("filename.txt", sep="\t", header=FALSE)`
- read.csv("filename.csv") # import data from csv file
- ironmansuits      <-      read.csv("C:/Users/evgen/Desktop/R_work_folder/Iron_Man_Suits.csv", stringsAsFactors=TRUE)
- mutate() # creates new columns or modifies current variables in the dataset
- diamonds %>% mutate(depth_times_price = depth * price)
- diamonds %>% mutate(depth_times_price = depth * price, is_cut_ideal = cut == "Ideal")
- analog of f-string in R: library(glue): `glue("Variable {varname}")`
- Export graph as .tiff: `ggsave("test.tiff", units="in", width=8, height=6, dpi=300, compression = 'lzw')`

## Operators AND (&), OR (|)

- diamonds %>% filter(cut == "Ideal", carat == "0.23", price > 400)
- select() # gives us specific columns
- diamonds %>% select (cut, price)
- arrange() # arranges the values in data frame by values in a variable
- diamonds %>% arrange(price)
- diamonds %>% arrange(desc(price))
- diamonds %>% arrange(price, carat)
- group_by() / summarise()
- diamonds %>% group_by(carat) %>% summarise(mean(price), count = n())
- diamonds %>% group_by(carat) %>% mutate(mean_price = mean(price))
- runif() # generates random deviates of uniform distribution

## Describing data

- str # describe the structure of dataset by type of variable
- mean, median, max, min, sd
- dim(my_data) ## show dimensions of a table
- dim(my_data)

- summary(my_data)
- summary(my_data$chrom)
- ?ggplot
- seq(1,10)
- data() # show a list of built-in datasets

## Datasets

```
# Create brand-new dataset
data <- data.frame(Chr = c("Chr1", "Chr2", "Chr2", "Chr3", "Chr4",
    "Chr11"), Type = c("1_Promoter", "1_Promoter", "2_Enhancer",
    "3_Activator", "1_Promoter", "3_Activator"), Value = c(1e+09,
    5e+10, 6e+09, 9e+10, 8e+10, 5e+10))
data$Chr = gsub("Chr", "", data$Chr)  # remove 'Chr' from column 1
data$Chr <- factor(data$Chr, levels = c(seq(1, 11), "X", "Y"))  # reorder the chromosomes numerically


# Create dataset from existing
diamonds2 <- tibble(diamonds$cut, diamonds$clarity)
names(diamonds2)[1:4] <- c("Cut", "Clarity")  # renames the first 4 columns
```

```
## Warning: The 'value' argument of 'names<-' must have the same length as 'x' as of tibble
## 3.0.0.
## i 'names' must have length 2, not 4.
```

```
# Add new columns
diamonds2 <- diamonds %>%
    mutate(price200 = price - 200, pricepercarat = price/carat)

# Update the in-table columns with new values
diamonds %>%
    mutate(cut = recode(cut, Ideal = "IDEAL", Good = "GOOD"))
```

## Filtering

- diamonds %>% filter(cut=="Fair" | cut=="Good", price <= 600)
- recode () # modifies the values within a variable
- filter () # gives us specific rows
- population2 <- population %>% filter(country == 'Brazil' | country == 'Argentina', year >= 2005)
- df$col1 == "name 1"
- df$col1 %in% c("name1", "name2")

## Data processing

- Rename df columns by name: `df.rename(columns={"A": "a", "B": "c"})`
- Rename df columns by index: `df.rename(index={0: "x", 1: "y", 2: "z"})`

Information below about tidy data is taken from this link.

**Dataset** - a collection of quantitative and qualitative values. Every value belongs to a variable and an observation.

**Untidy (messy) data** - rows and columns are mixed up with observations and variables.

**Tidy data** - a standard way of mapping the meaning of a dataset to its structure. Variables are in columns and observations - in rows. Tidy data makes it easy to extract variables.

**Examples**:

Untidy (messy) data - 6 observations but only 3 rows - 'wide' dataset:

```r
# define data
df <- data.frame(. = c("John", "Jane", "Mary"), Treatment_A = c(16,
    3, 2), Treatment_B = c(0, 4, 5), Treatment_C = c(0, 8, 10))
df
```

```
##      . Treatment_A Treatment_B Treatment_C
## 1 John          16           0           0
## 2 Jane           3           4           8
## 3 Mary           2           5          10
```

Tidy data structure - three variables - Person/name, Treatment, and Result - 'long/tall' dataset - made by melting the untidy one:

```r
df <- data.frame(`Person/name` = c("John", "Jane", "Mary", "John",
    "Jane", "Mary", "John", "Jane", "Mary"), Treatment = c("a",
    "a", "a", "b", "b", "b", "c", "c", "c"), Result = c(16, 3,
    2, 0, 4, 5, 0, 8, 10))
df
```

```
##   Person.name Treatment Result
## 1        John         a     16
## 2        Jane         a      3
## 3        Mary         a      2
## 4        John         b      0
## 5        Jane         b      4
## 6        Mary         b      5
## 7        John         c      0
## 8        Jane         c      8
## 9        Mary         c     10
```

## Data Visualisations

By convention, independent variable is on X axis, and dependent variable is on Y axis.

### Dotplots

**Dataset 'population'**

```r
# Filtering a sample dataset 'population'
population2 <- population %>%
    filter(country == "Brazil" | country == "Argentina", year >=
        2000)
# population2

# Simple dotplot
ggplot(population2, aes(x = year, y = population, color = country)) +
    geom_point(size = 4) + scale_x_continuous(breaks = (seq(min(population2$year),
    max(population2$year), by = 3))) + theme(panel.grid.major = element_blank(),
    panel.grid.minor = element_blank(), axis.text = element_text(size = 13),
    axis.title = element_text(size = 15), panel.background = element_blank(),
    axis.line = element_line(size = 1)) + scale_color_brewer(palette = "Set1") +
    labs(title = "Population growth in Brazil and Argentina",
        x = "Year", y = "Population")
```

```
## Warning: The 'size' argument of 'element_line()' is deprecated as of ggplot2 3.4.0.
## i Please use the 'linewidth' argument instead.
```
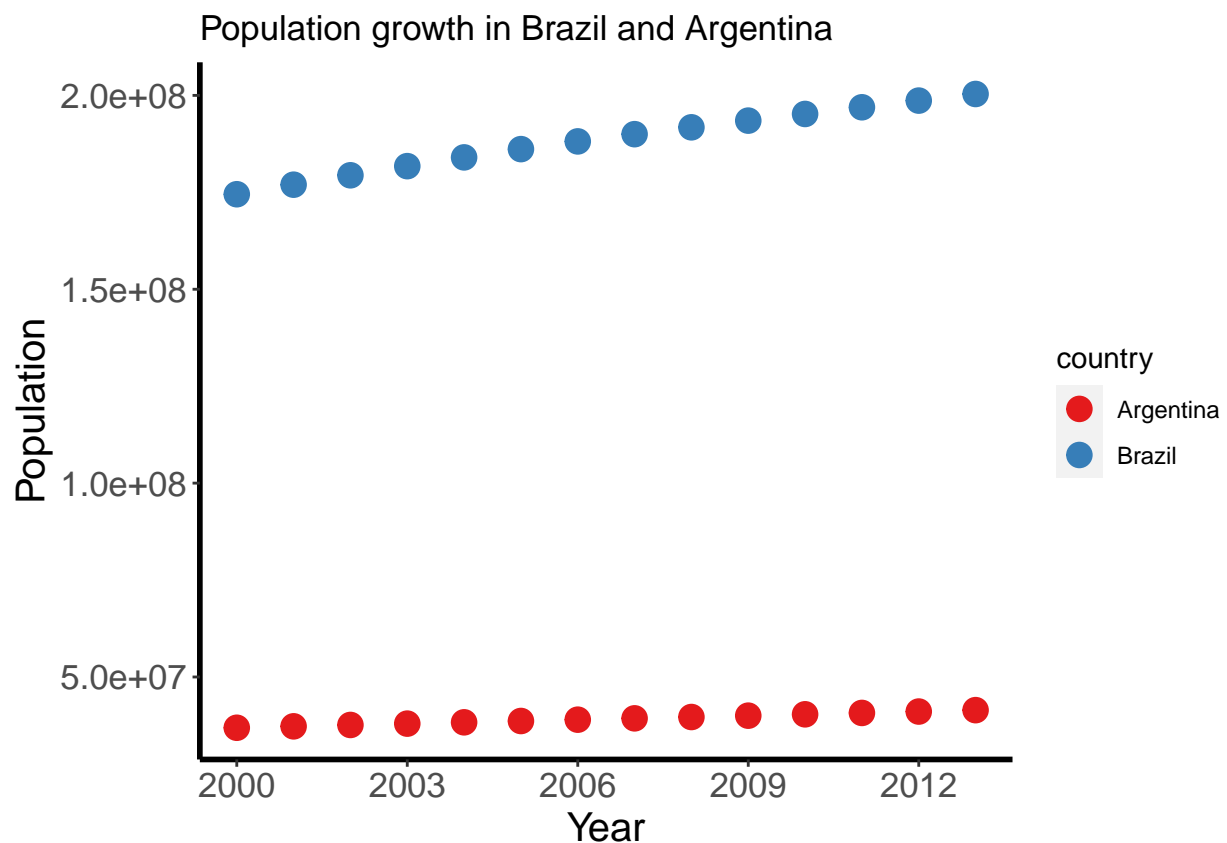


```r
# Faceted dotplot
ggplot(population2, aes(x = year, y = population)) + geom_point() +
    facet_wrap(~country, nrow = 2)
```
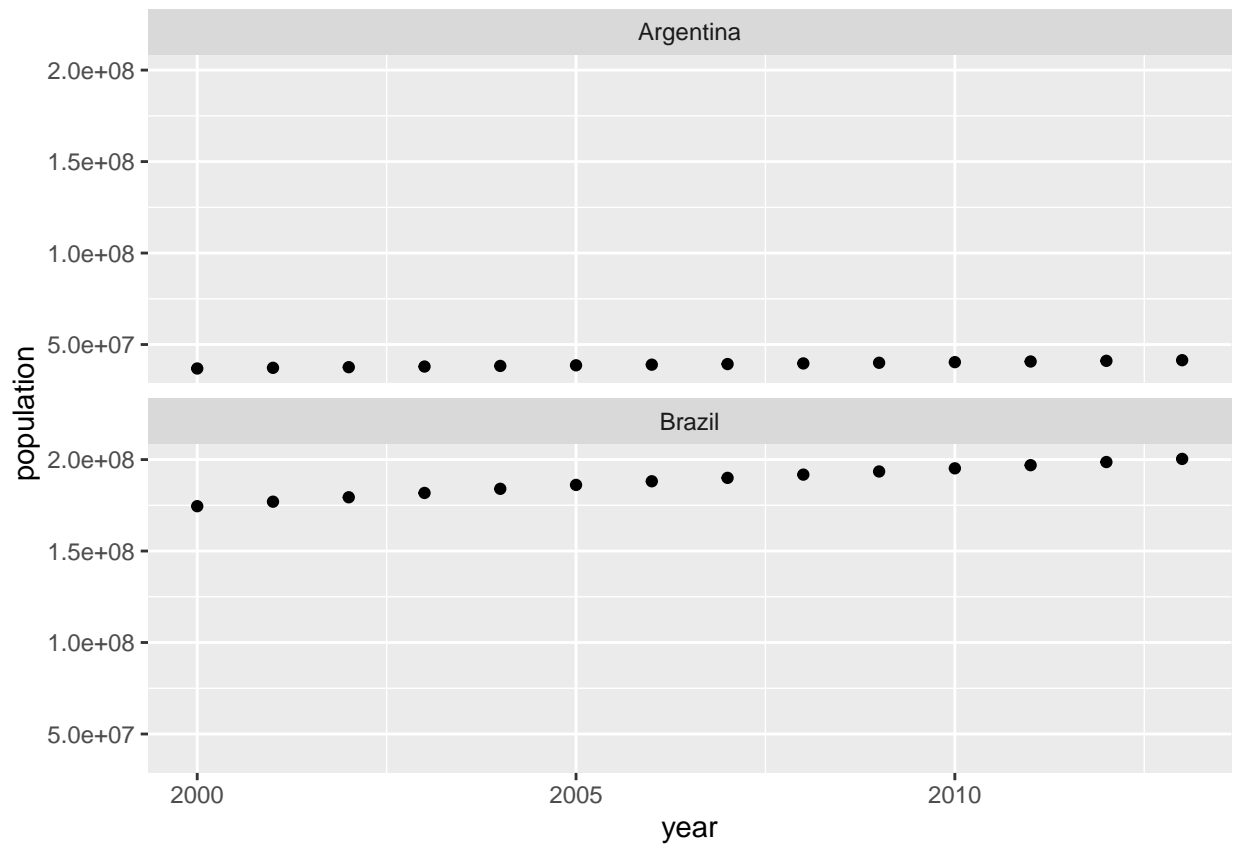
```
# Dotplot with callouts
ggplot(population2, aes(x = year, y = population, color = country,
    label = population)) + geom_text(check_overlap = TRUE) +
    geom_label()
```
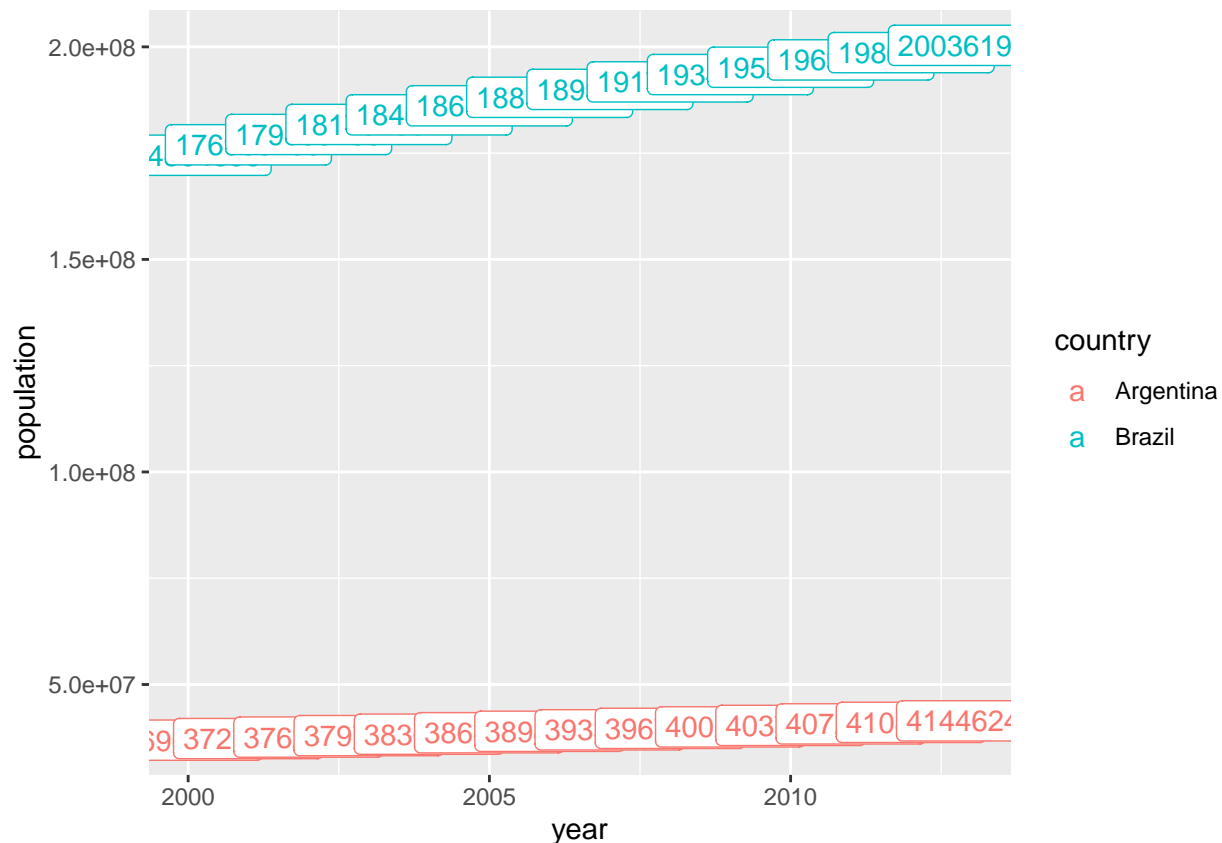
## Scatterplots

```
msleep
```

```
## # A tibble: 83 x 11
##     name          genus vore  order conse~1 sleep~2 sleep~3 sleep~4 awake  brainwt
##     <chr>         <chr> <chr> <chr> <chr>     <dbl>   <dbl>   <dbl> <dbl>    <dbl>
##  1 Cheetah       Acin~ carni Carn~ lc         12.1    NA    NA      11.9 NA
##  2 Owl monkey    Aotus omni  Prim~ <NA>       17       1.8  NA       7    0.0155
##  3 Mountain be~  Aplo~ herbi Rode~ nt         14.4     2.4  NA       9.6 NA
##  4 Greater sho~  Blar~ omni  Sori~ lc         14.9     2.3   0.133   9.1  0.00029
##  5 Cow           Bos   herbi Arti~ domest~     4       0.7   0.667  20    0.423
##  6 Three-toed ~  Brad~ herbi Pilo~ <NA>       14.4     2.2   0.767   9.6 NA
##  7 Northern fu~  Call~ carni Carn~ vu          8.7     1.4   0.383  15.3 NA
##  8 Vesper mouse  Calo~ <NA>  Rode~ <NA>        7       NA    NA      17   NA
##  9 Dog           Canis carni Carn~ domest~    10.1     2.9   0.333  13.9  0.07
## 10 Roe deer      Capr~ herbi Arti~ lc          3       NA    NA      21    0.0982
## # ... with 73 more rows, 1 more variable: bodywt <dbl>, and abbreviated
## #   variable names 1: conservation, 2: sleep_total, 3: sleep_rem,
## #   4: sleep_cycle
```

```
# Scatterplot with df msleep
ggplot(msleep, aes(x = bodywt, y = sleep_total, colour = vore)) +
    geom_point() + xlim(0, 7000) + ylim(0, 21)
```
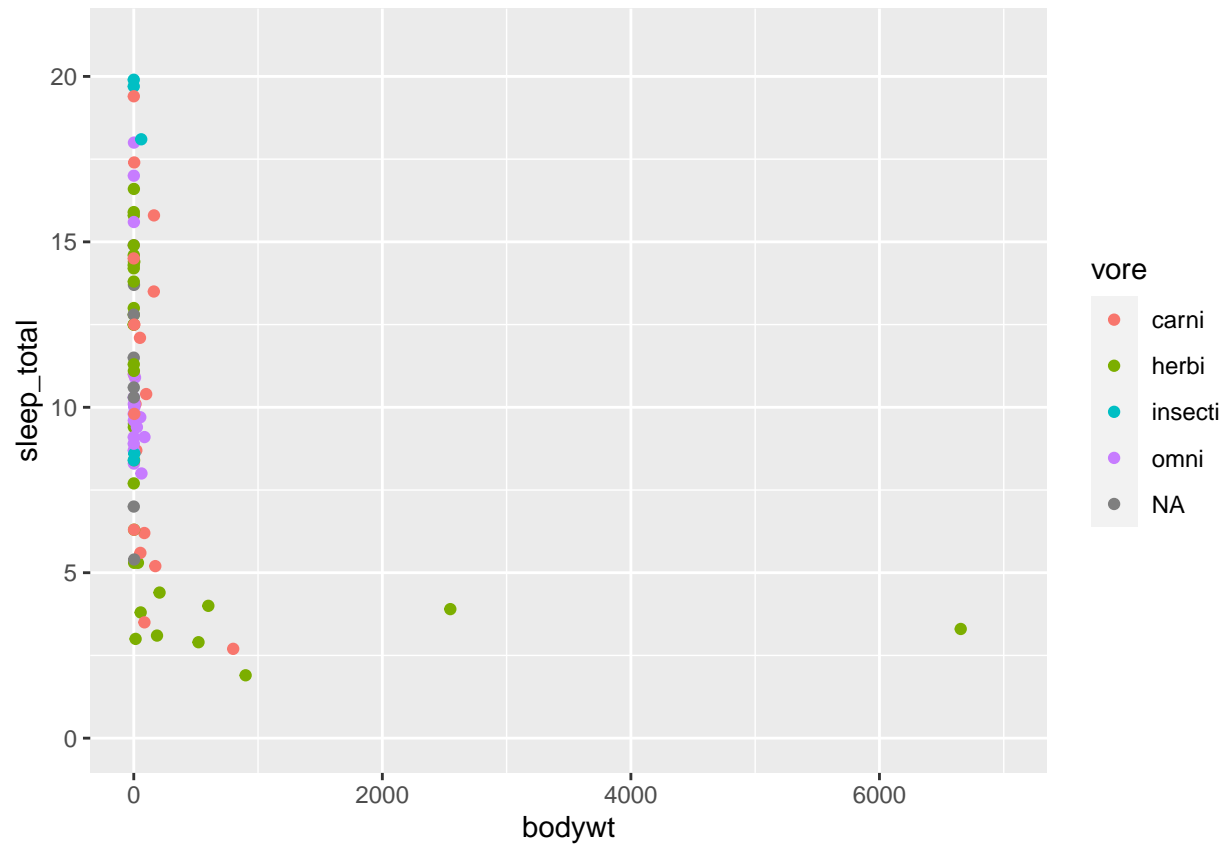
## Histograms

```
ggplot(diamonds, aes(x = price)) + geom_bar()
```
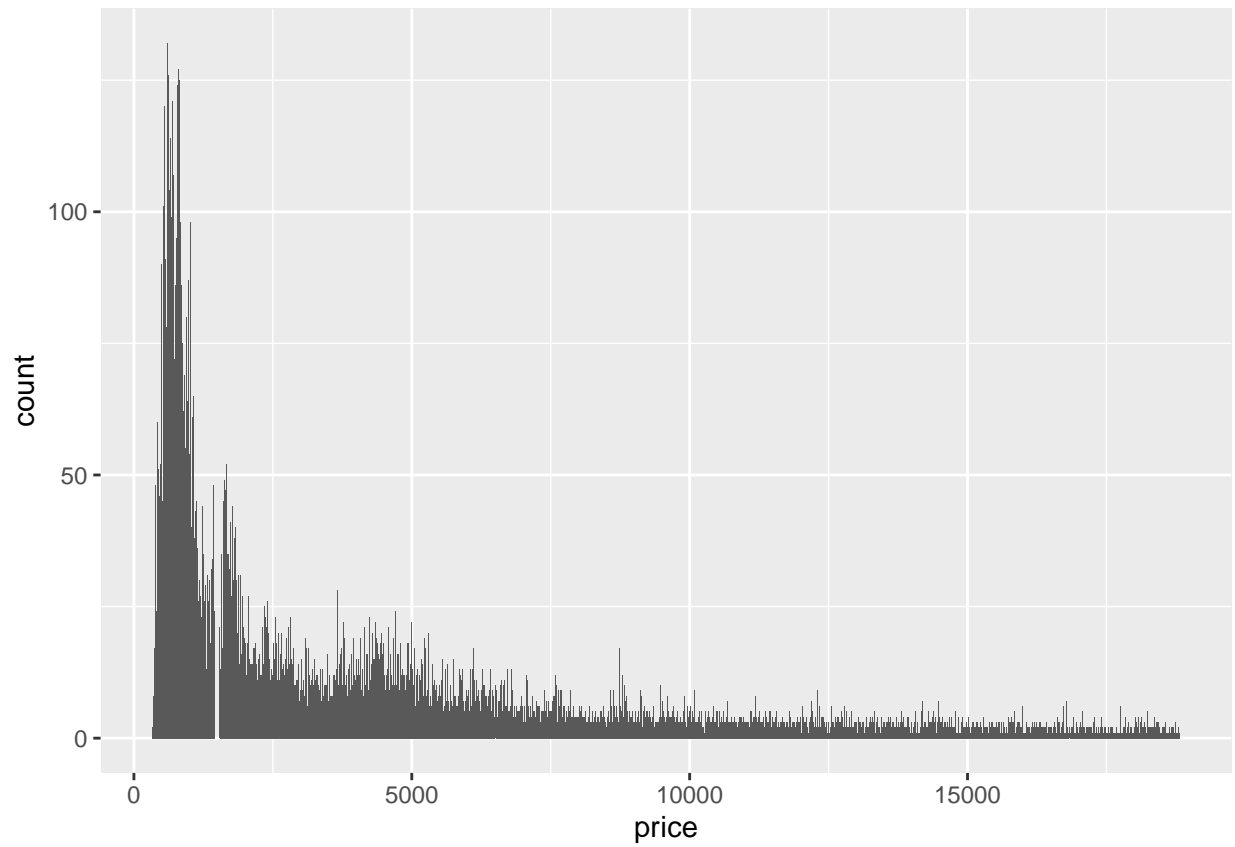
```
ggplot(diamonds, aes(x = price, fill = cut)) + geom_bar()
```
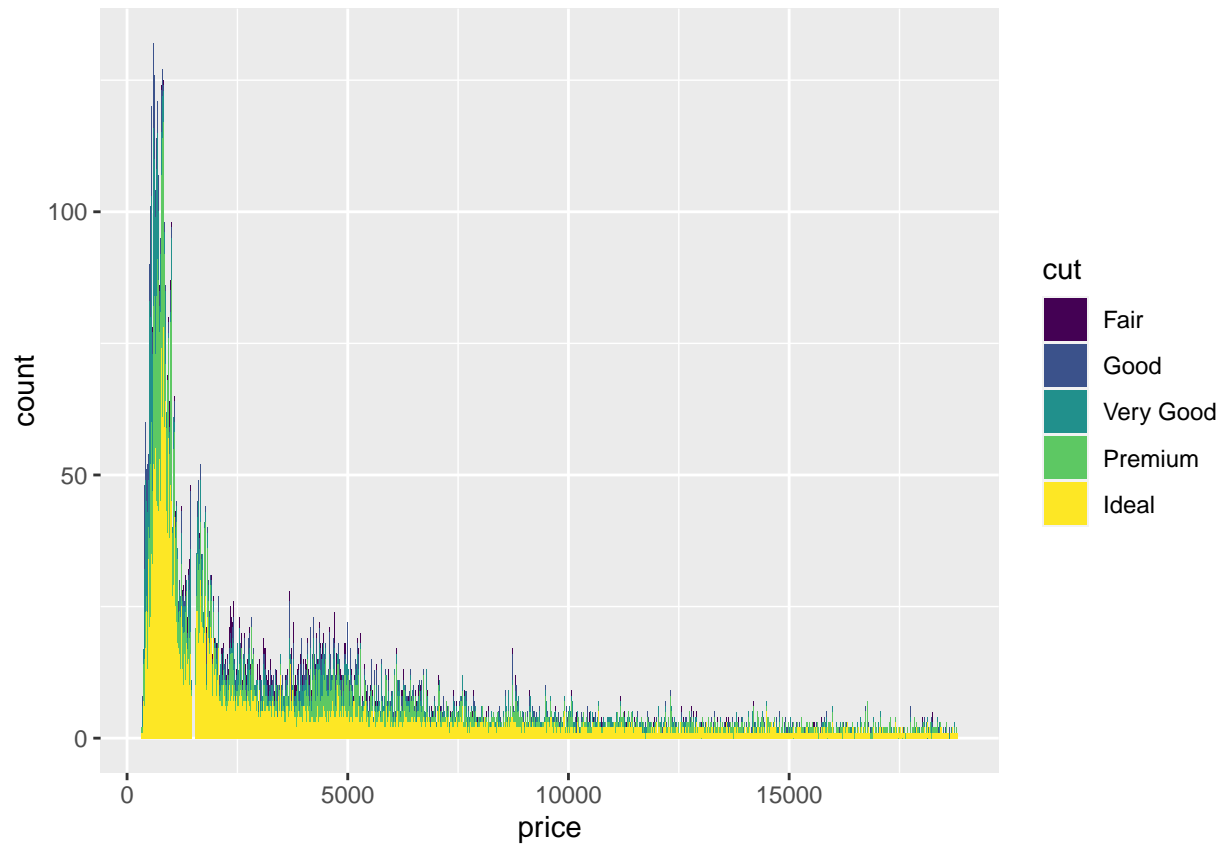
```
ggplot(diamonds, aes(x = price, fill = cut)) + geom_histogram(binwidth = 500) +
    xlim(0, 10000)
```

## Warning: Removed 5222 rows containing non-finite values ('stat_bin()').

## Warning: Removed 10 rows containing missing values ('geom_bar()').
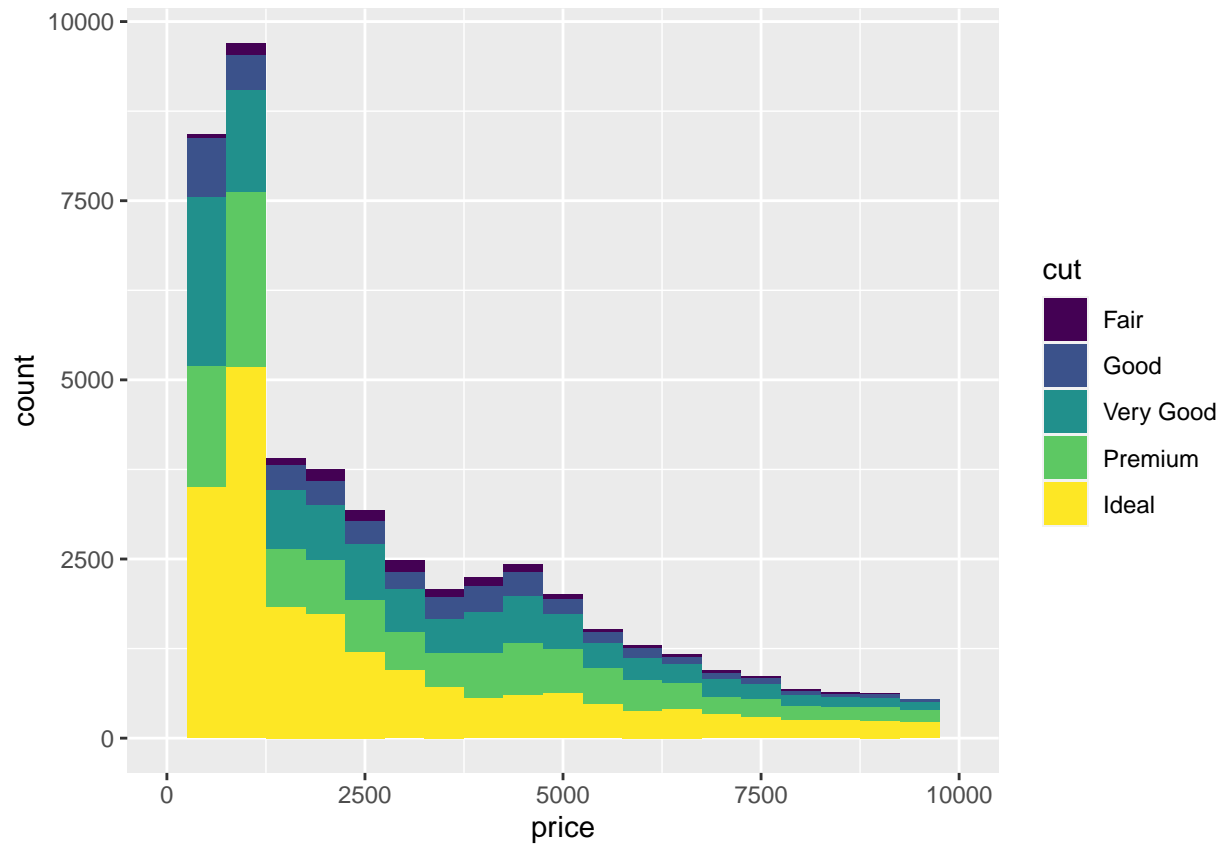
## Bar graphs

For geom_bar, aggregate numbers of rows for each x-value can be counted by default with argument stat=count. Alternatively, the values can be provided manually for the bar graph by stat="identity".

```
ggplot(diamonds, aes(x = cut)) + geom_bar()
```
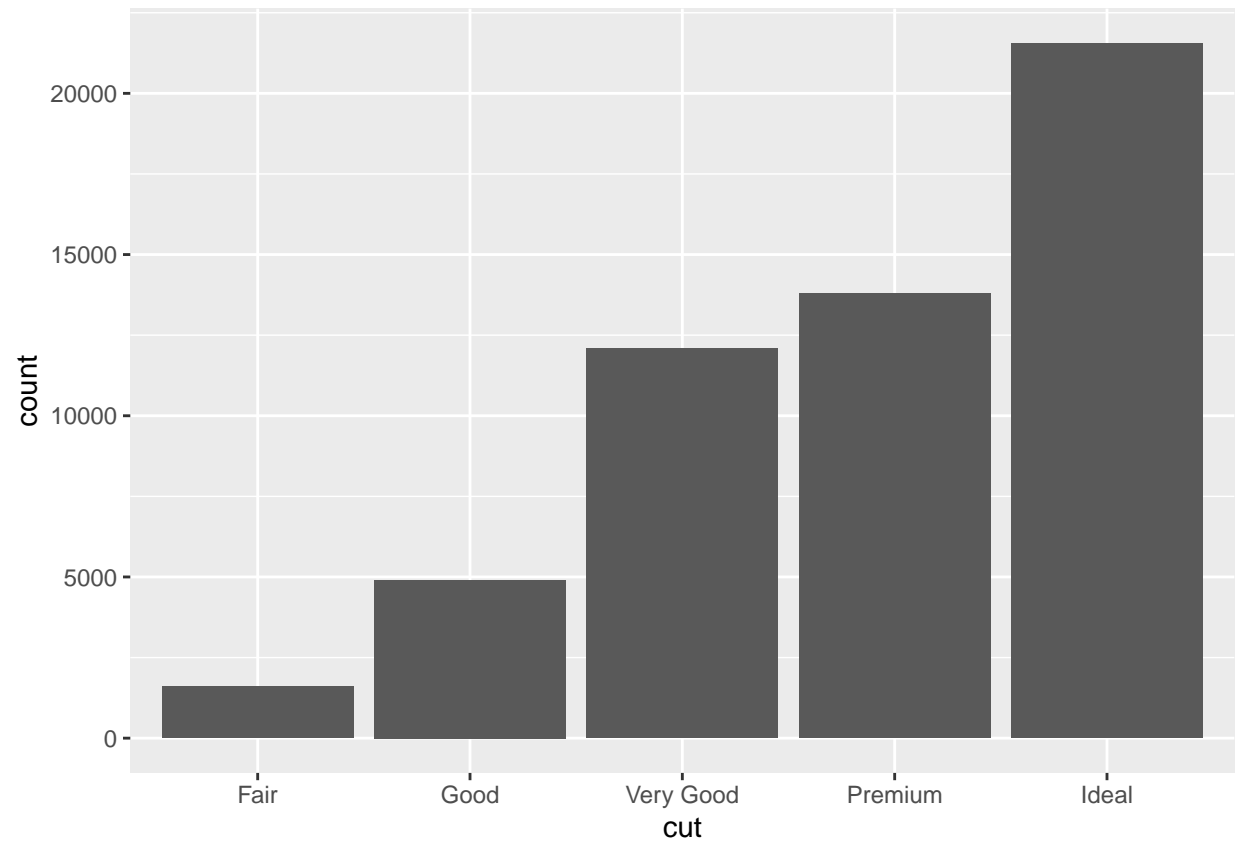
```
ggplot(diamonds, aes(x = cut, y = price)) + geom_bar(stat = "identity")
```
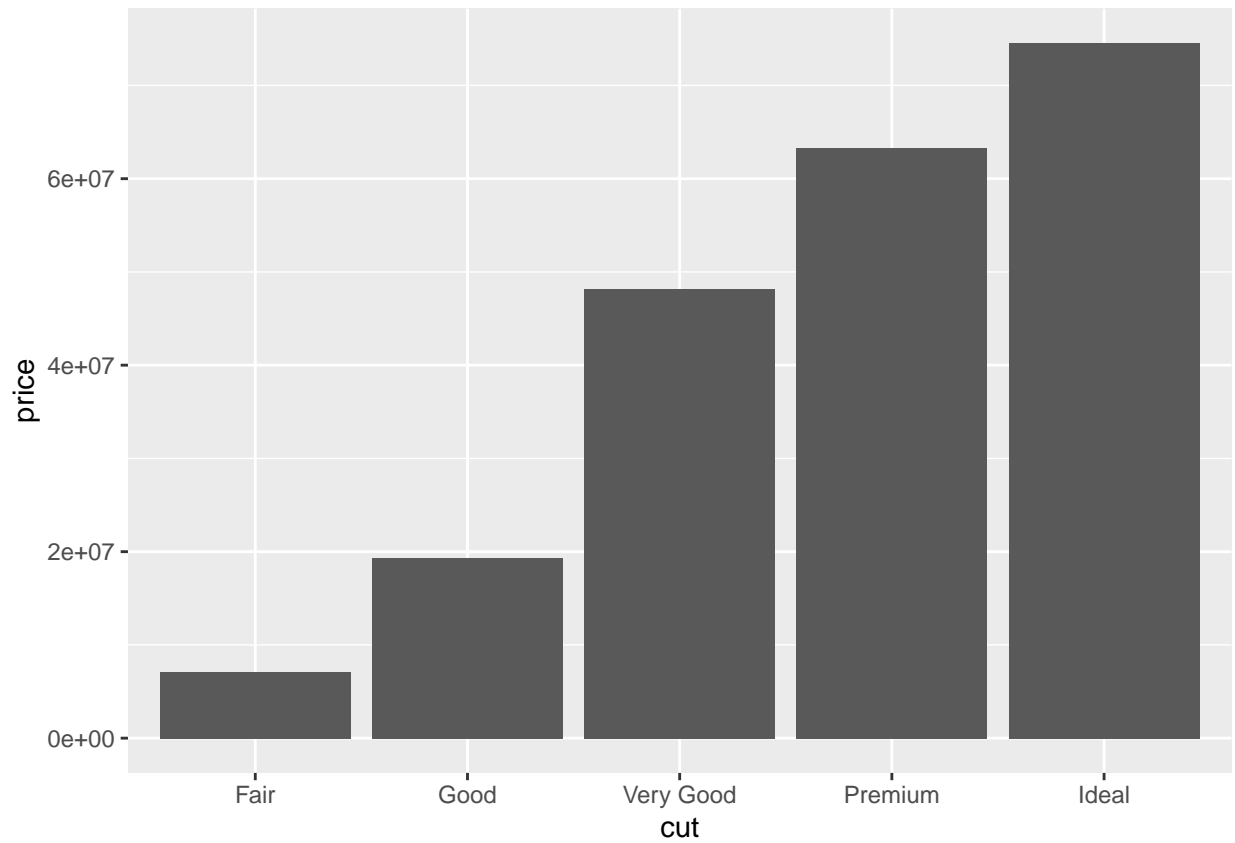
```r
ggplot(diamonds, aes(x = cut, fill = cut)) + geom_bar() + geom_text(stat = "count",
    aes(label = ..count..), vjust = 1.5, size = 3.5, colour = "white")
```

```
## Warning: The dot-dot notation ('..count..') was deprecated in ggplot2 3.4.0.
## i Please use 'after_stat(count)' instead.
```
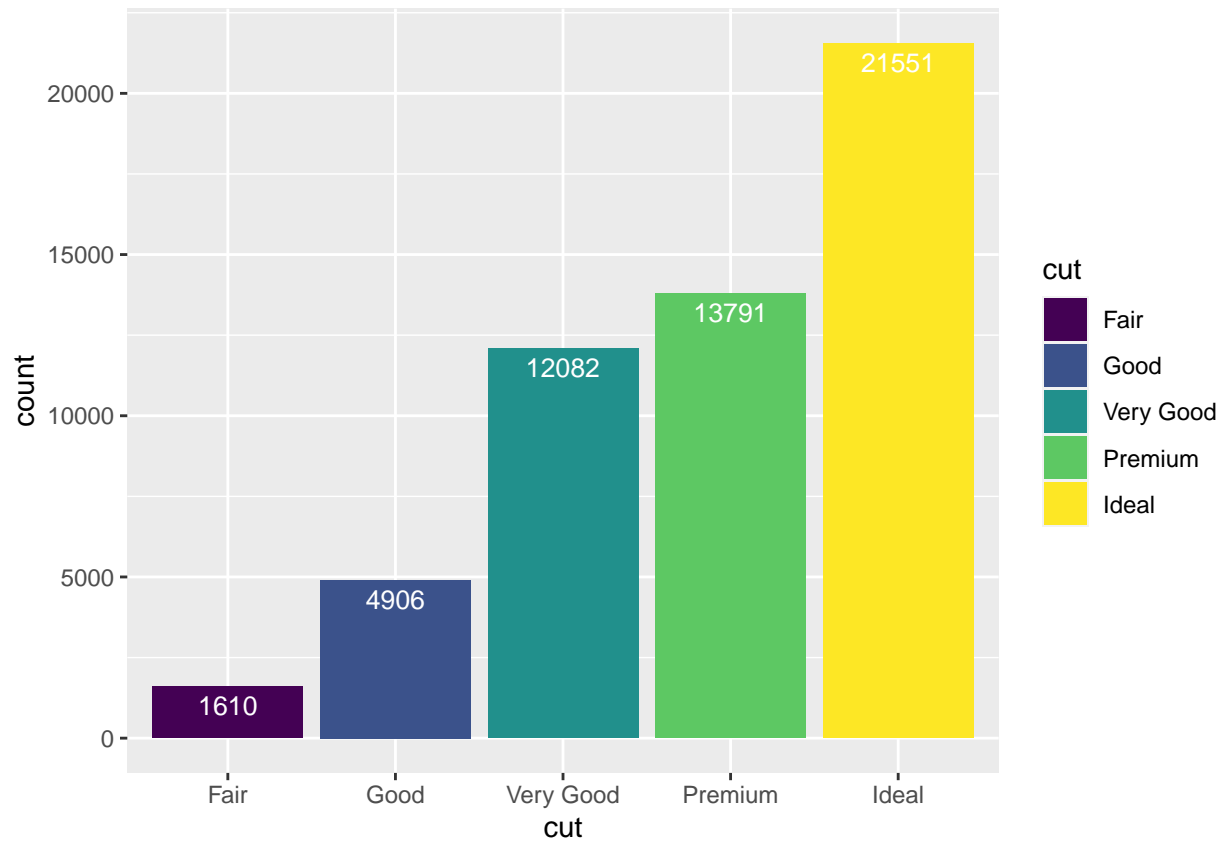
```
ggplot(diamonds, aes(x = cut, y = price, fill = cut)) + geom_col()
```
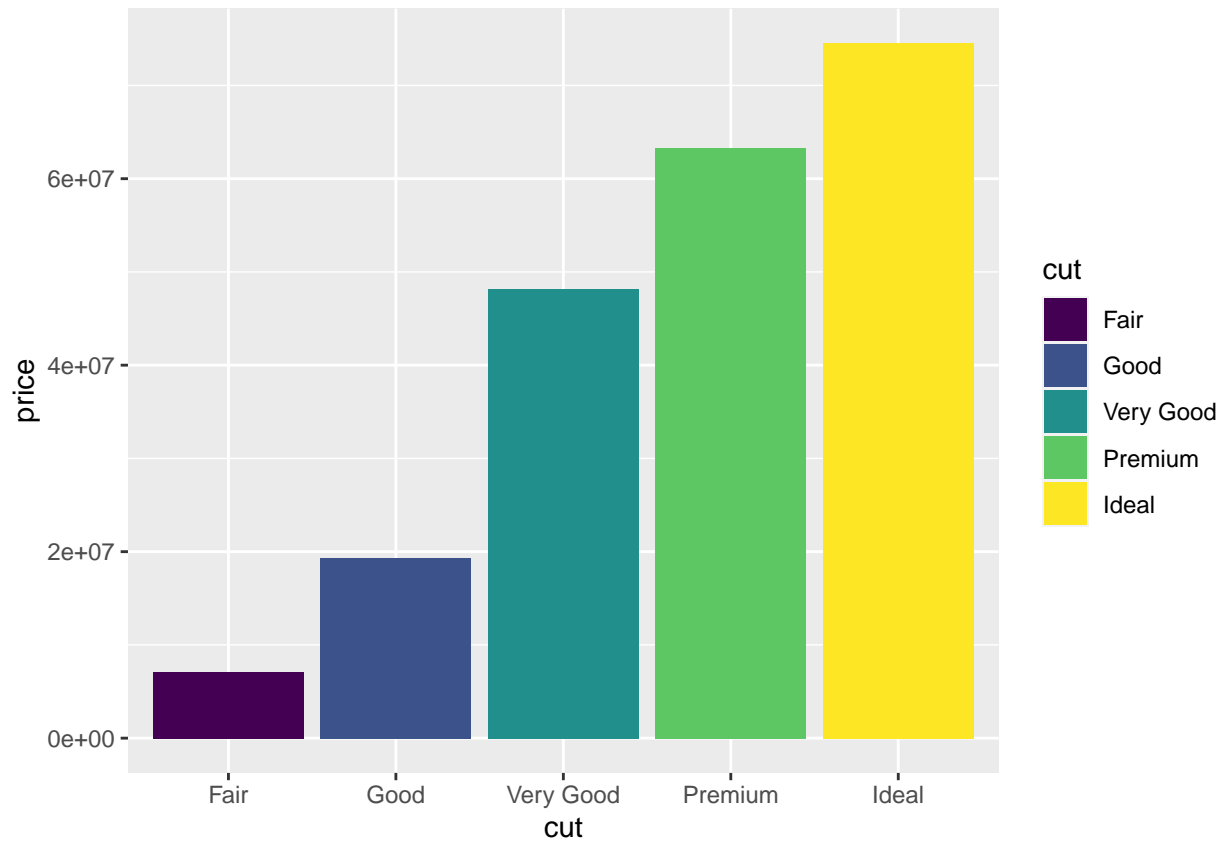
## Stacked component bar graphs

```
ggplot(diamonds, aes(x=cut, fill=color)) +
  geom_bar()
```
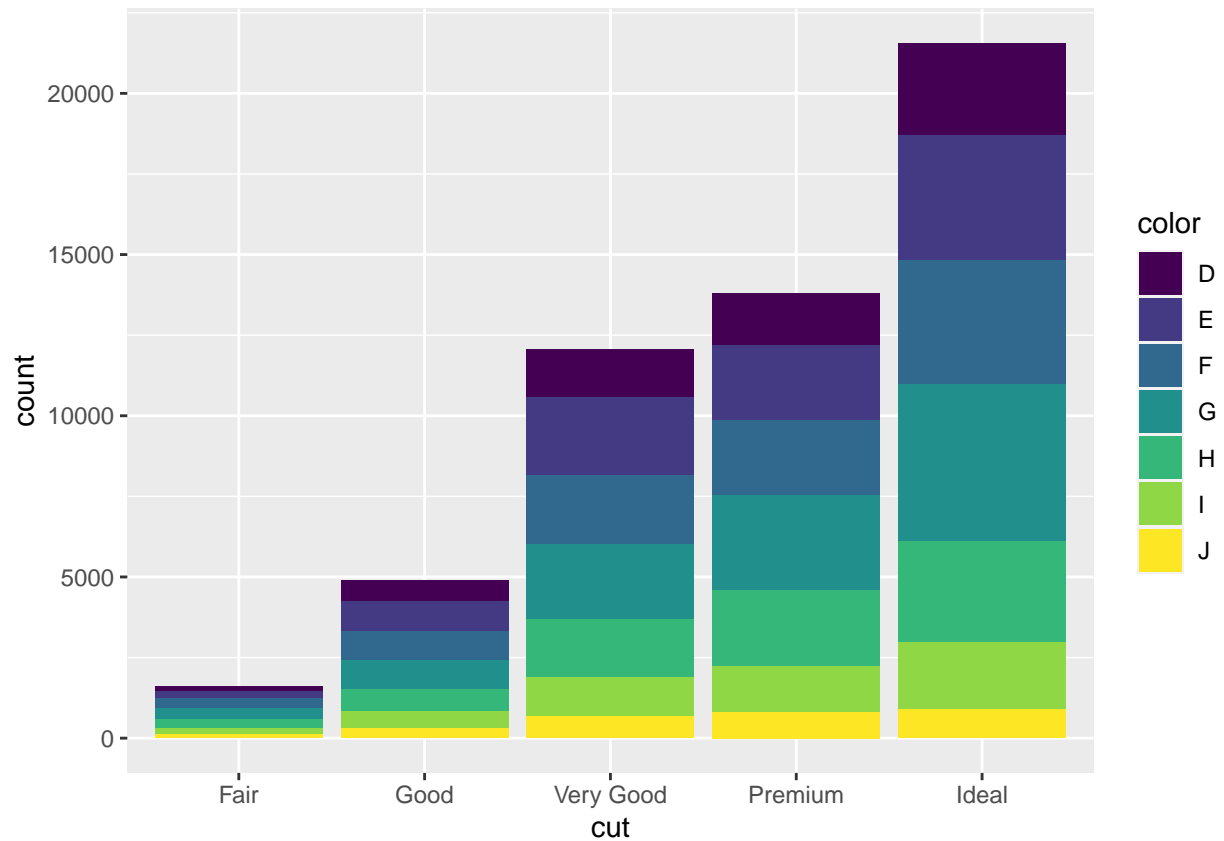
```
starwars %>%
  filter(hair_color == "black" |          # | = OR
          hair_color == "brown") %>%
  drop_na(sex) %>%
  ggplot(aes(x = hair_color, fill = sex)) +
  geom_bar(alpha = 0.7) +
  theme_bw() +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank()) +
  labs(title = "Gender and Hair Colour",
       x = "Hair Colour",
       y = "Number")
```

## Gender and Hair Colour
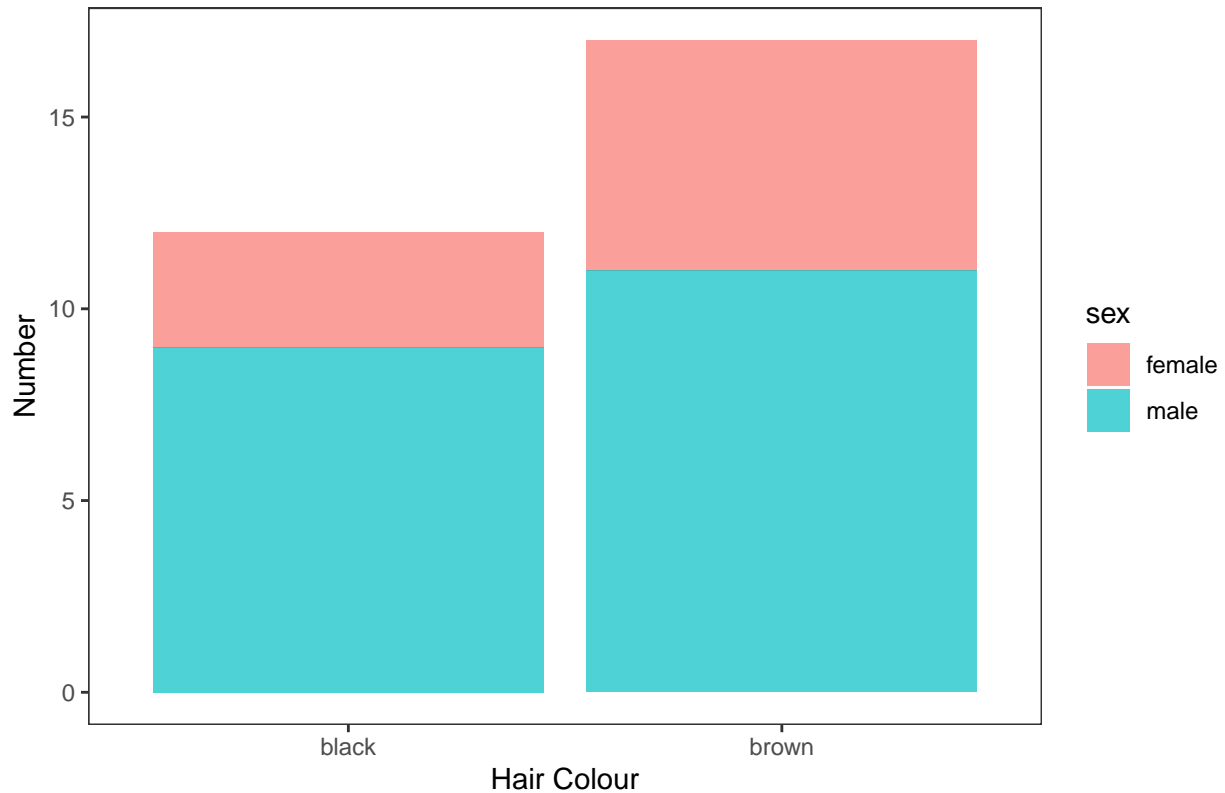


## Stacked percentage bar graphs

```
library(reshape)

# Example 1
starwars
```

```
## # A tibble: 87 x 14
##     name        height  mass hair_~1 skin_~2 eye_c~3 birth~4 sex    gender homew~5
##     <chr>        <int> <dbl> <chr>   <chr>   <chr>     <dbl> <chr>  <chr>  <chr>
##  1 Luke Skywa~    172    77 blond   fair    blue         19 male   mascu~ Tatooi~
##  2 C-3PO          167    75 <NA>    gold    yellow      112 none   mascu~ Tatooi~
##  3 R2-D2           96    32 <NA>    white,~ red          33 none   mascu~ Naboo
##  4 Darth Vader    202   136 none    white   yellow     41.9 male   mascu~ Tatooi~
##  5 Leia Organa    150    49 brown   light   brown        19 fema~  femin~ Aldera~
##  6 Owen Lars      178   120 brown,~ light   blue         52 male   mascu~ Tatooi~
##  7 Beru White~    165    75 brown   light   blue         47 fema~  femin~ Tatooi~
##  8 R5-D4           97    32 <NA>    white,~ red          NA none   mascu~ Tatooi~
##  9 Biggs Dark~    183    84 black   light   brown        24 male   mascu~ Tatooi~
## 10 Obi-Wan Ke~    182    77 auburn~ fair    blue-g~      57 male   mascu~ Stewjon
## # ... with 77 more rows, 4 more variables: species <chr>, films <list>,
## #   vehicles <list>, starships <list>, and abbreviated variable names
## #   1: hair_color, 2: skin_color, 3: eye_color, 4: birth_year, 5: homeworld
```

```
starwars %>%
  filter(hair_color == "black" |          # | = OR
           hair_color == "brown") %>%
  drop_na(sex) %>%
  ggplot(aes(x = sex, fill = hair_color)) +
  geom_bar(position = "fill",
           alpha = 0.5) +
  theme_bw() +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank()) +
  labs(title = "Gender and Hair Colour",
       x = "Sex",
       y = "Percentage")
```
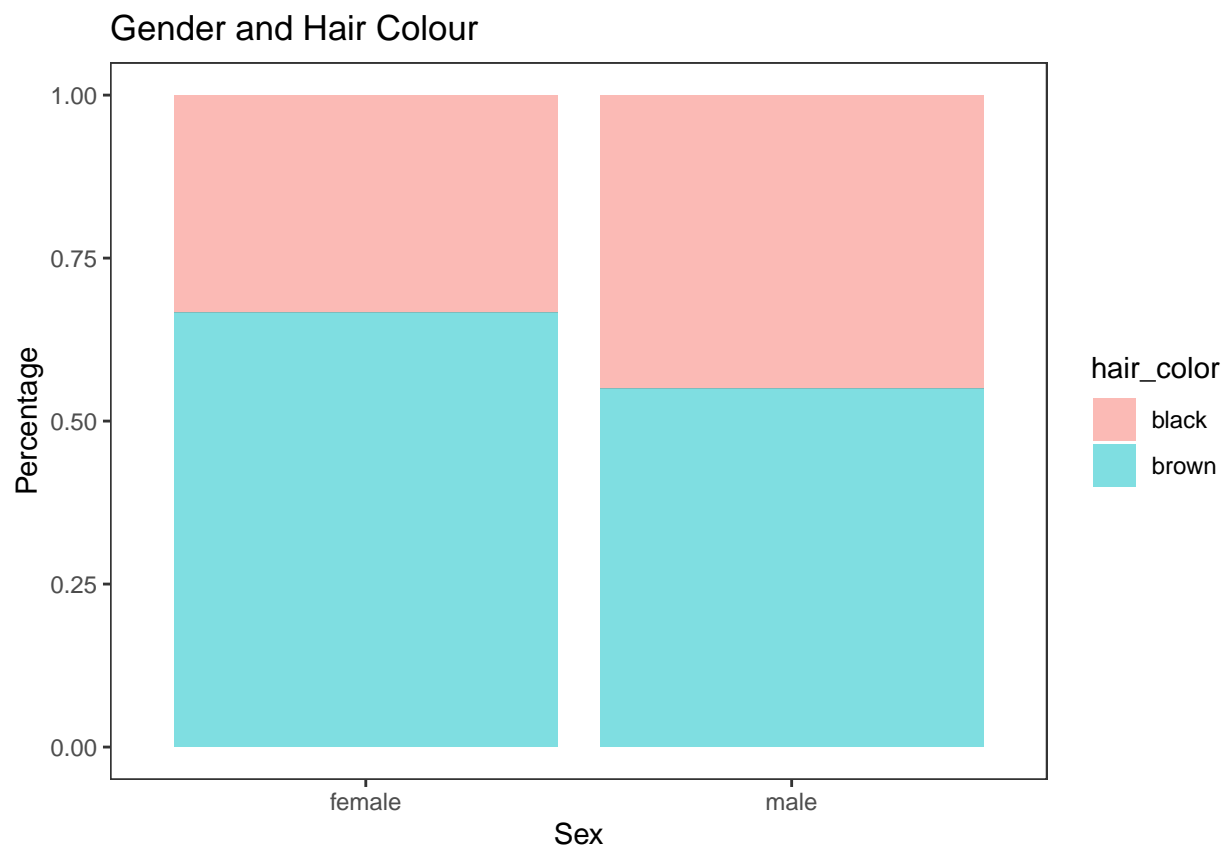


```
# Example 2
relig_income
```

```
## # A tibble: 18 x 11
##    religion       '<$10k' $10-2~1 $20-3~2 $30-4~3 $40-5~4 $50-7~5 $75-1~6 $100-~7
##    <chr>            <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>
##  1 Agnostic            27      34      60      81      76     137     122     109
##  2 Atheist             12      27      37      52      35      70      73      59
##  3 Buddhist            27      21      30      34      33      58      62      39
##  4 Catholic           418     617     732     670     638    1116     949     792
##  5 Don't know/r~       15      14      15      11      10      35      21      17
```

```
##  6 Evangelical ~       575     869    1064     982     881    1486     949     723
##  7 Hindu                 1       9       7       9      11      34      47      48
##  8 Historically~       228     244     236     238     197     223     131      81
##  9 Jehovah's Wi~        20      27      24      24      21      30      15      11
## 10 Jewish               19      19      25      25      30      95      69      87
## 11 Mainline Prot       289     495     619     655     651    1107     939     753
## 12 Mormon               29      40      48      51      56     112      85      49
## 13 Muslim                6       7       9      10       9      23      16       8
## 14 Orthodox             13      17      23      32      32      47      38      42
## 15 Other Christ~         9       7      11      13      13      14      18      14
## 16 Other Faiths         20      33      40      46      49      63      46      40
## 17 Other World ~         5       2       3       4       2       7       3       4
## 18 Unaffiliated        217     299     374     365     341     528     407     321
## # ... with 2 more variables: '>150k' <dbl>, 'Don't know/refused' <dbl>, and
## #   abbreviated variable names 1: '$10-20k', 2: '$20-30k', 3: '$30-40k',
## #   4: '$40-50k', 5: '$50-75k', 6: '$75-100k', 7: '$100-150k'
```

```r
relig_income %>%
  filter(religion == "Atheist" |
          religion == "Mormon" |
          religion == "Buddhist") %>%
  melt(id.vars = "religion") %>%
  group_by(religion, variable) %>%
  ggplot(aes(x = religion, y = value, fill = variable)) +
  geom_bar(stat="identity", position = "fill") +
  theme_bw() +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        axis.title = element_text()) +
  scale_color_brewer(palette = "Set1") +
  labs(title = "Distribution of income levels by religion",
       x = "Religion",
       y = "Percentage")
```

# Distribution of income levels by religion
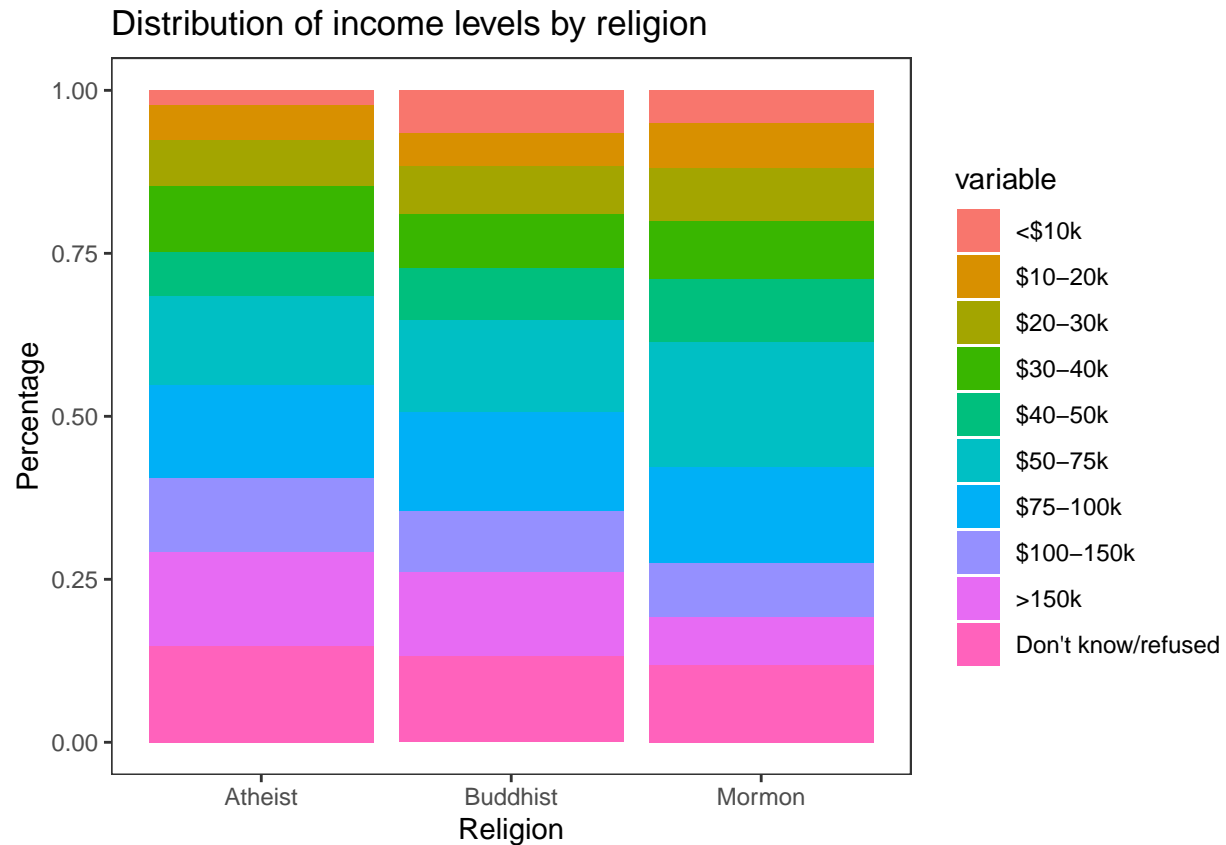


```
# library gcookbook
cabbage_exp
```

```
##   Cultivar Date Weight        sd  n         se
## 1      c39  d16   3.18 0.9566144 10 0.30250803
## 2      c39  d20   2.80 0.2788867 10 0.08819171
## 3      c39  d21   2.74 0.9834181 10 0.31098410
## 4      c52  d16   2.26 0.4452215 10 0.14079141
## 5      c52  d20   3.11 0.7908505 10 0.25008887
## 6      c52  d21   1.47 0.2110819 10 0.06674995
```

```
ggplot(cabbage_exp, aes(x = Date, y = Weight, fill = Cultivar)) +
  geom_col(position = "fill", width=0.5) +
  scale_y_continuous(labels = scales::percent) +
  theme_bw() +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank()) +
    scale_fill_lancet() + # module ggsci - palettes from sci journals
  labs(title = "Title",
      x = "date",
      y = "Weight, kg")
```
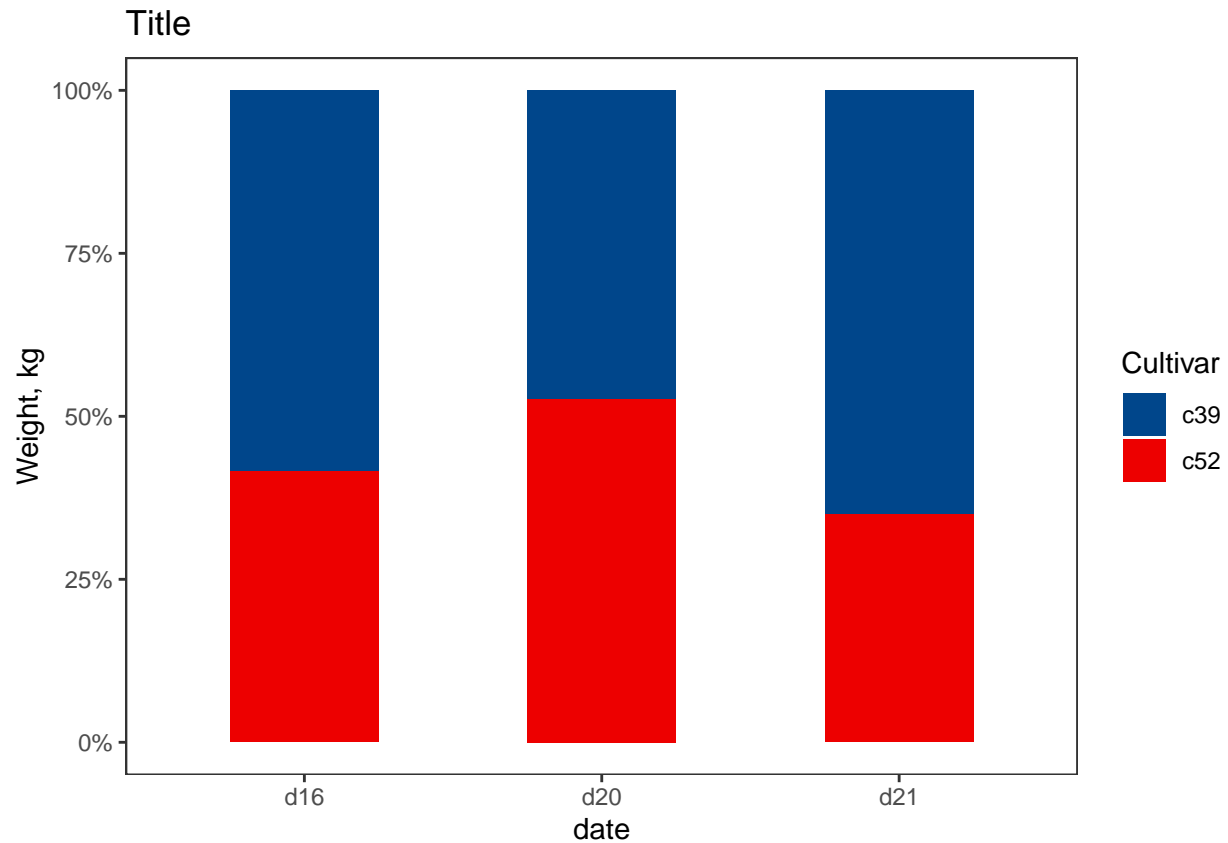
```
# Custom dataset
df <- data.frame (Type_of_epistasis = c('negative', 'positive', 'negative', 'positive'),
                  n = c(250, 750, 300, 700),
                  algorithm = c('algorithm1', 'algorithm1', 'algorithm2', 'algorithm2'),
                  percentage = c(0.25, 0.75, 0.3, 0.7),
                  percentage_100 = c(25, 75, 30, 70))
df
```

```
##   Type_of_epistasis   n  algorithm percentage percentage_100
## 1          negative 250 algorithm1       0.25             25
## 2          positive 750 algorithm1       0.75             75
## 3          negative 300 algorithm2       0.30             30
## 4          positive 700 algorithm2       0.70             70
```

```
ggplot(df, aes(x=algorithm, y=percentage, fill=Type_of_epistasis)) +
  geom_col(position='fill', colour='black') +
  scale_y_continuous(labels = scales::percent) +
  geom_text(aes(label = percentage_100), size = 4, position=position_stack(vjust=0.5), color='white') +
  theme_bw() +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank()) +
  scale_fill_lancet() + # module ggsci - palettes from sci journals
  labs(title = "Distributions of epistasis between two algorithms",
       x = "Algorithm",
       y = "Distribution of types of epistasis, %")
```

Distributions of epistasis between two algorithms
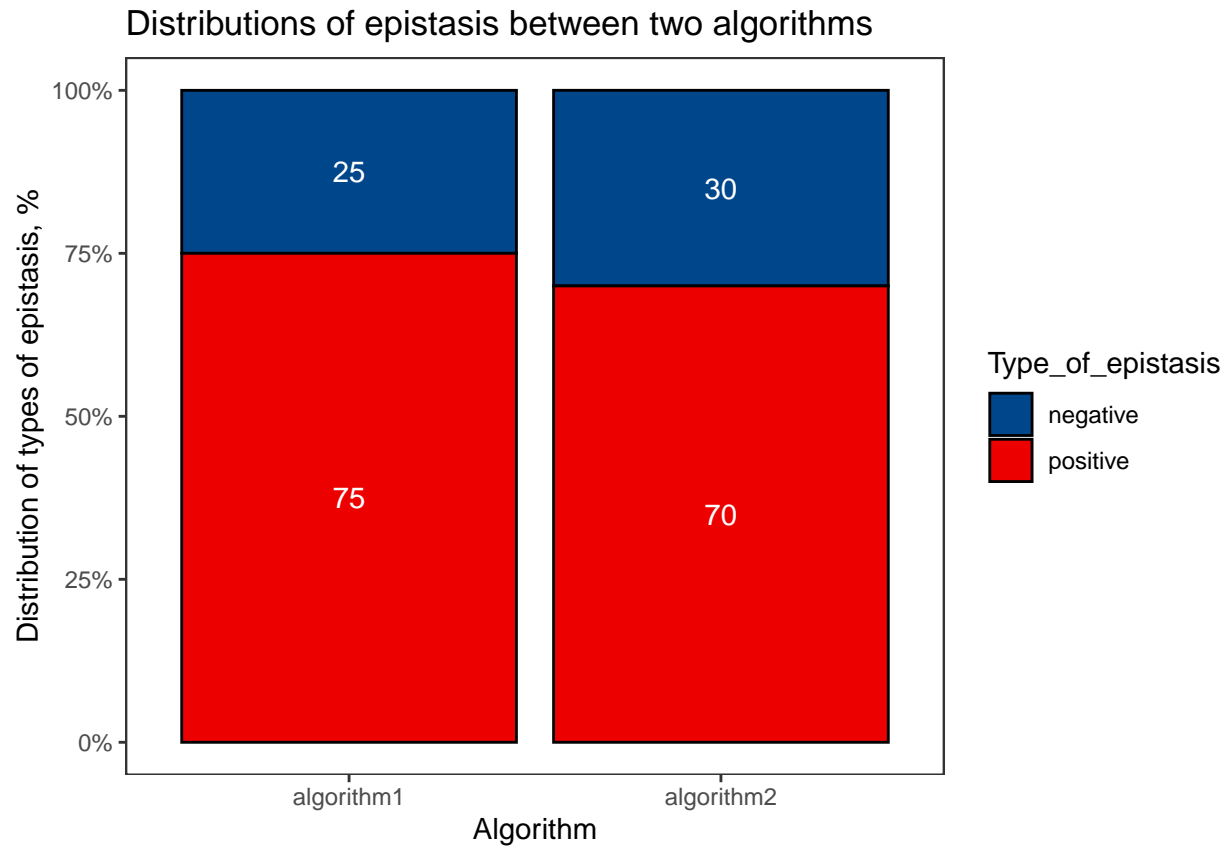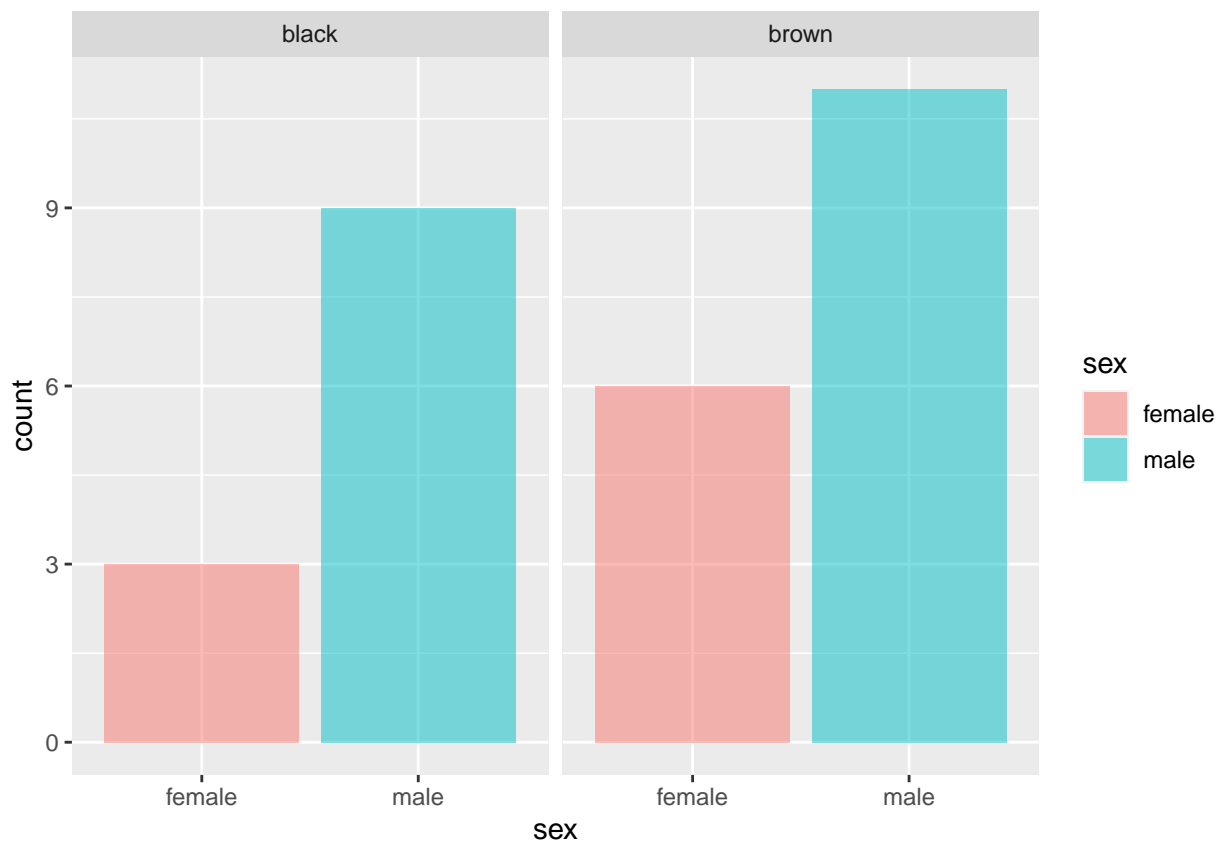
## Grouped bar graphs

```
ggplot(diamonds, aes(x = cut, fill = clarity)) + geom_bar(position = "dodge")
```
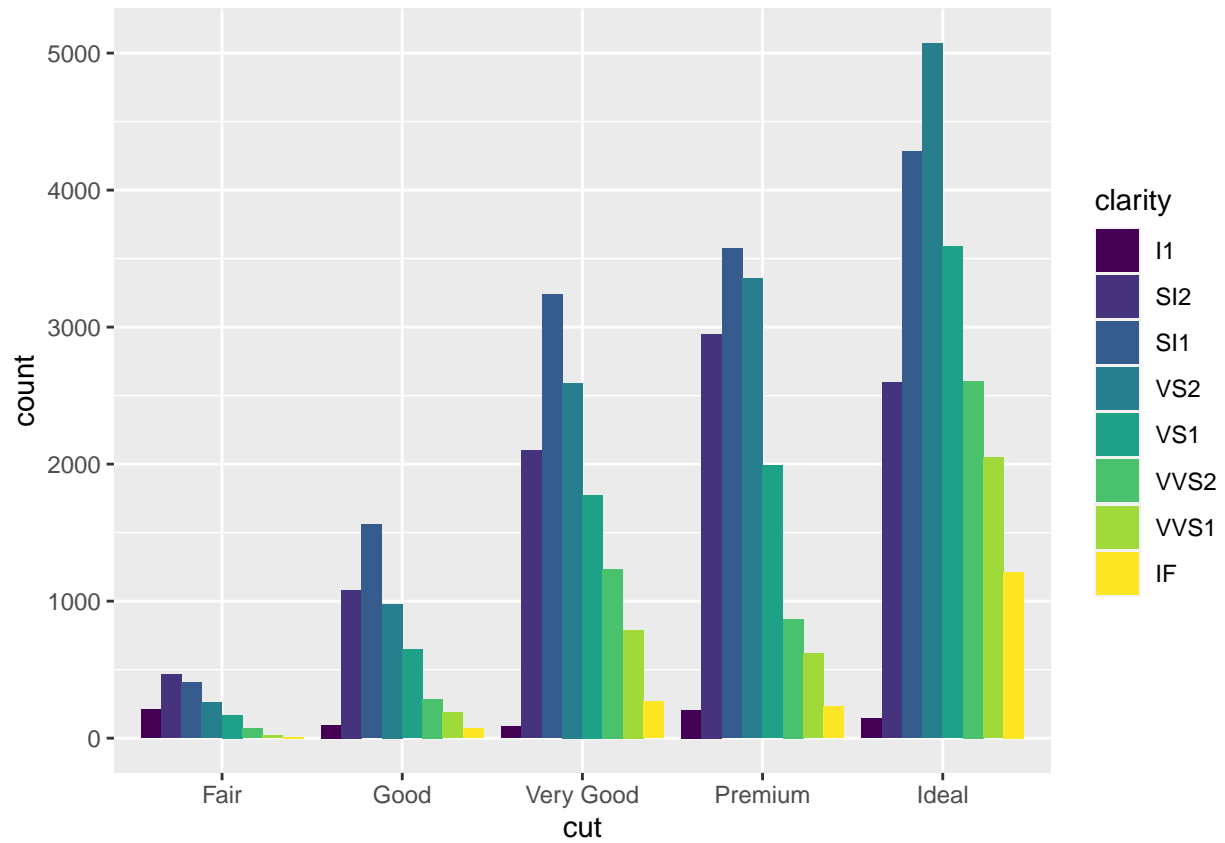
```
# grouped facet
starwars %>%
    filter(hair_color %in% c("black", "brown")) %>%
    drop_na(sex) %>%
    ggplot(aes(sex)) + geom_bar(aes(fill = sex), alpha = 0.5) +
    facet_wrap(~hair_color)
```

```
# grouped bar graphs with error bars
head(ToothGrowth)
```

```
##    len supp dose
## 1  4.2   VC  0.5
## 2 11.5   VC  0.5
## 3  7.3   VC  0.5
## 4  5.8   VC  0.5
## 5  6.4   VC  0.5
## 6 10.0   VC  0.5
```

```
df <- ToothGrowth
df$dose <- as.factor(df$dose)
data_summary <- function(data, varname, groupnames) {
    require(plyr)
    summary_func <- function(x, col) {
        c(mean = mean(x[[col]], na.rm = TRUE), sd = sd(x[[col]],
            na.rm = TRUE))
    }
    data_sum <- ddply(data, groupnames, .fun = summary_func,
        varname)
    data_sum <- rename(data_sum, c(mean = varname))
    return(data_sum)
}
df2 <- data_summary(ToothGrowth, varname = "len", groupnames = c("supp",
    "dose"))
```

```
## Loading required package: plyr

## --------------------------------------------------------------------------------

## You have loaded plyr after dplyr - this is likely to cause problems.
## If you need functions from both plyr and dplyr, please load plyr first, then dplyr:
## library(plyr); library(dplyr)

## --------------------------------------------------------------------------------

##
## Attaching package: 'plyr'

## The following objects are masked from 'package:plotly':
##
##     arrange, mutate, rename, summarise

## The following object is masked from 'package:ggpubr':
##
##     mutate

## The following objects are masked from 'package:reshape':
##
##     rename, round_any

## The following objects are masked from 'package:dplyr':
##
##     arrange, count, desc, failwith, id, mutate, rename, summarise,
##     summarize

## The following object is masked from 'package:purrr':
##
##     compact
```

```r
## Convert dose to a factor variable
df2$dose = as.factor(df2$dose)
head(df2)
```

```
##   supp dose   len       sd
## 1   OJ  0.5 13.23 4.459709
## 2   OJ    1 22.70 3.910953
## 3   OJ    2 26.06 2.655058
## 4   VC  0.5  7.98 2.746634
## 5   VC    1 16.77 2.515309
## 6   VC    2 26.14 4.797731
```

```r
p <- ggplot(df2, aes(x = dose, y = len, fill = supp)) + geom_bar(stat = "identity",
    color = "black", position = position_dodge()) + geom_errorbar(aes(ymin = len -
    sd, ymax = len + sd), width = 0.2, position = position_dodge(0.9))
print(p)
```
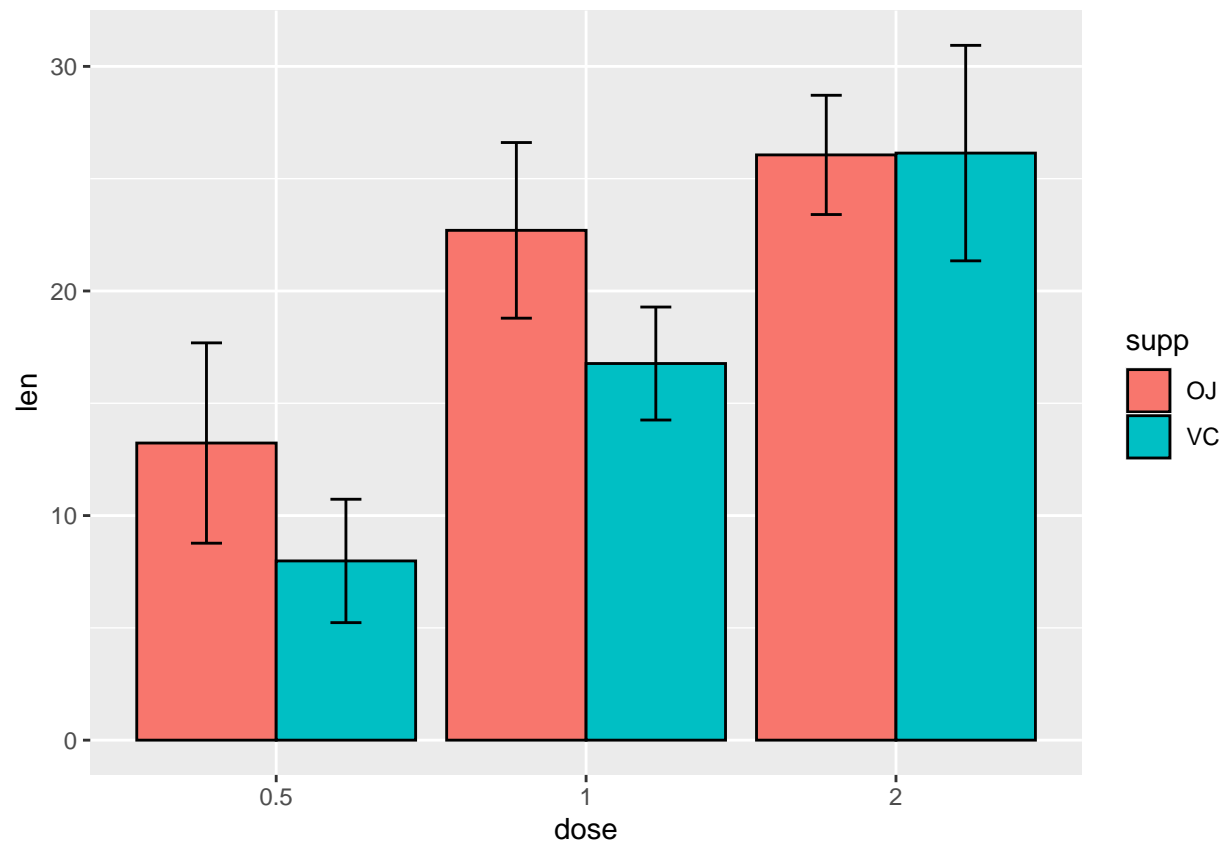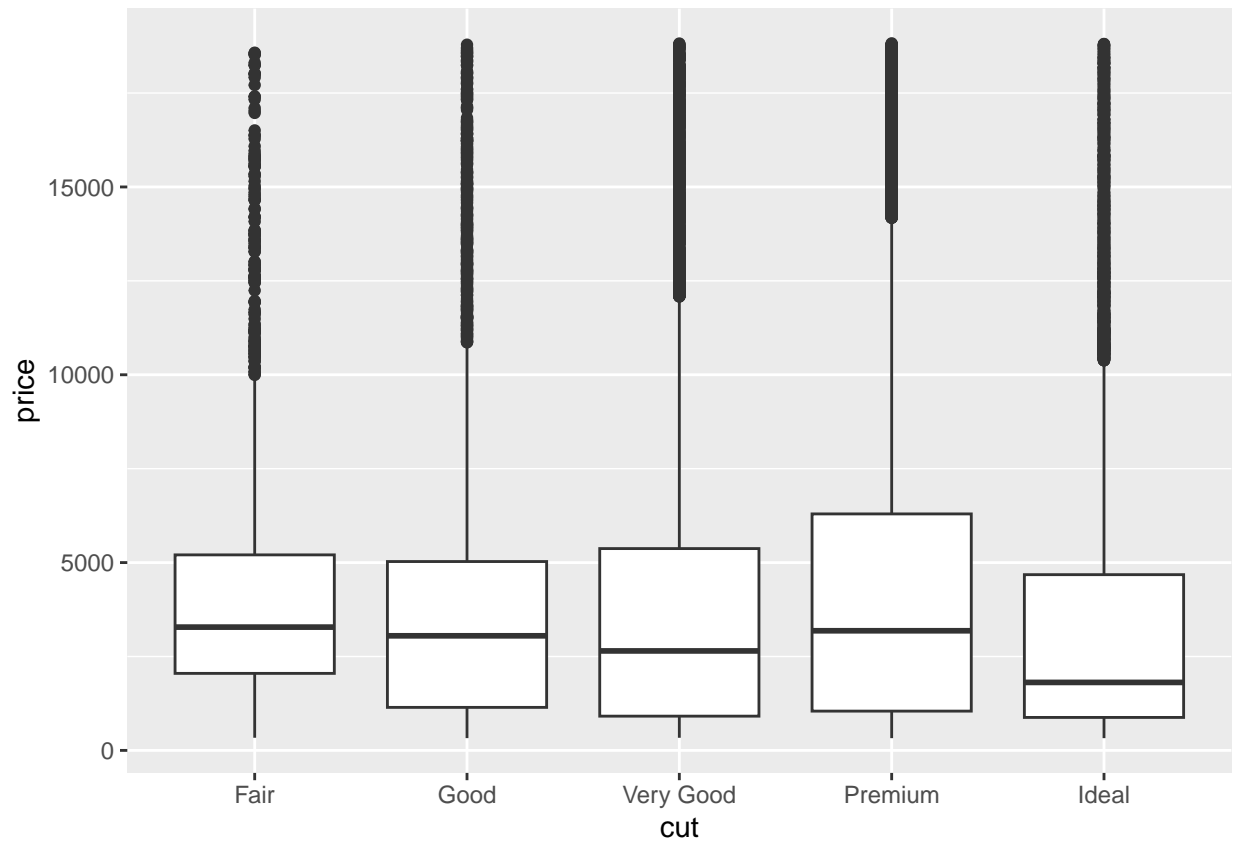
```
## Finished bar plot
p + labs(title = "Tooth length per dose", x = "Dose (mg)", y = "Length") +
    theme_classic() + scale_fill_manual(values = c("#999999",
    "#E69F00"))
```
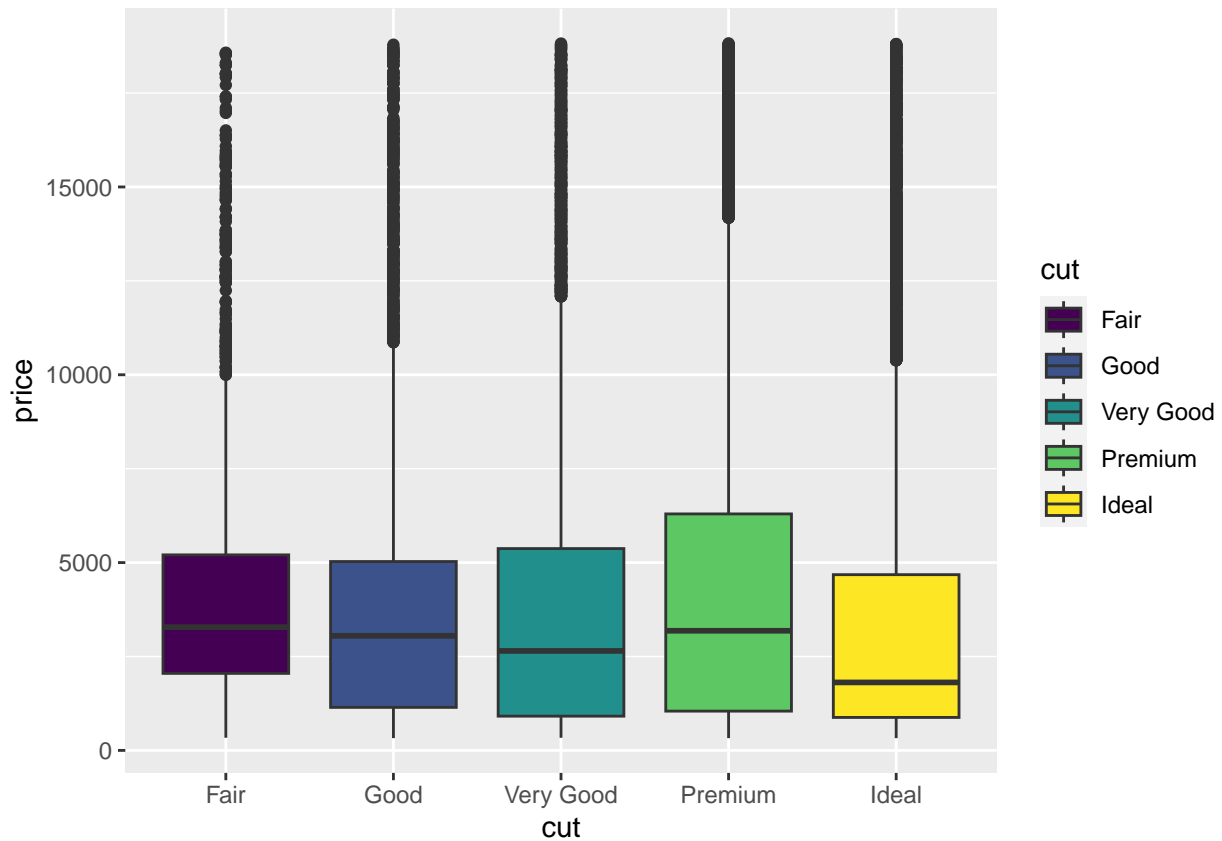
Tooth length per dose
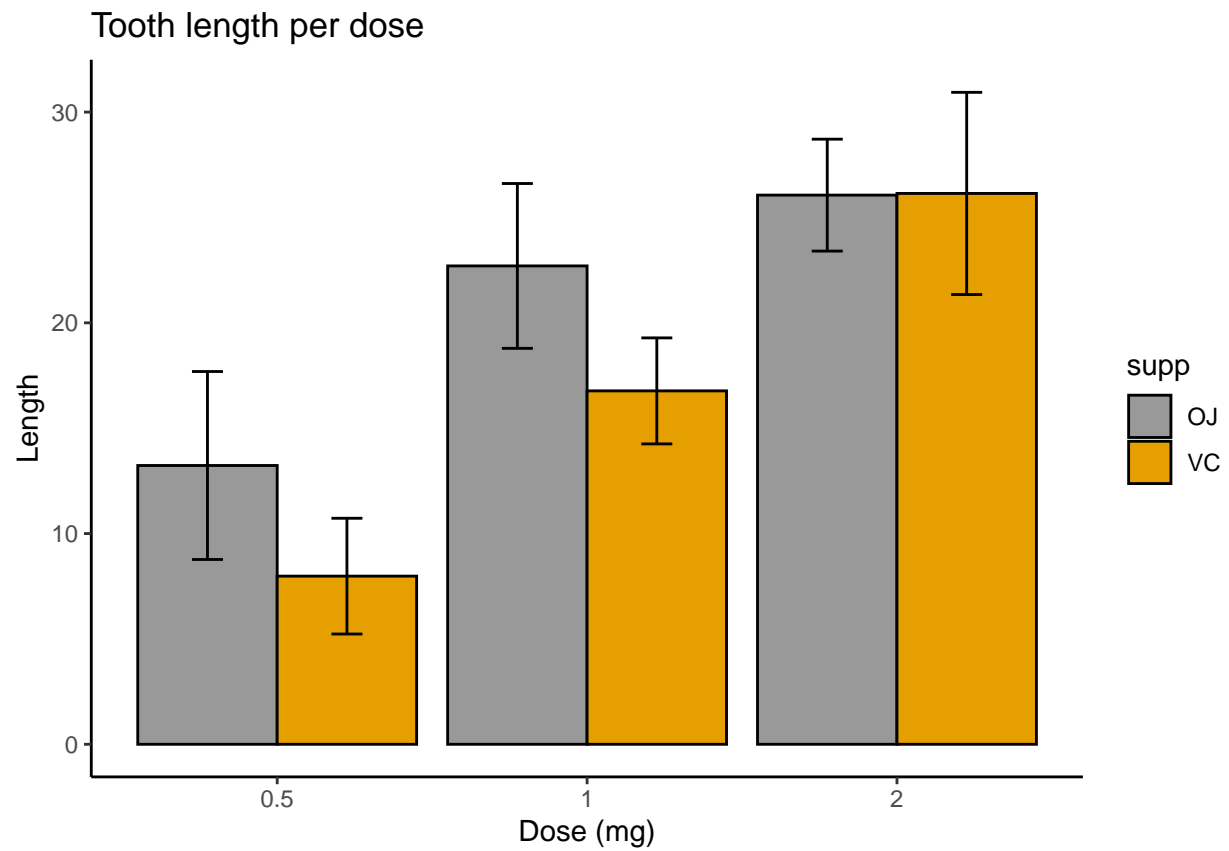
## Boxplots

```
ggplot(diamonds, aes(x = cut, y = price)) + geom_boxplot()
```

```
ggplot(diamonds, aes(x = cut, y = price, fill = cut)) + geom_boxplot()
```

## Violin plots

```
starwars
```

```
## # A tibble: 87 x 14
##    name         height  mass hair_~1 skin_~2 eye_c~3 birth~4 sex    gender homew~5
##    <chr>         <int> <dbl> <chr>   <chr>   <chr>     <dbl> <chr>  <chr>  <chr>
##  1 Luke Skywa~     172    77 blond   fair    blue         19 male   mascu~ Tatooi~
##  2 C-3PO           167    75 <NA>    gold    yellow      112 none   mascu~ Tatooi~
##  3 R2-D2            96    32 <NA>    white,~ red          33 none   mascu~ Naboo
##  4 Darth Vader     202   136 none    white   yellow     41.9 male   mascu~ Tatooi~
##  5 Leia Organa     150    49 brown   light   brown        19 fema~  femin~ Aldera~
##  6 Owen Lars       178   120 brown,~ light   blue         52 male   mascu~ Tatooi~
##  7 Beru White~     165    75 brown   light   blue         47 fema~  femin~ Tatooi~
##  8 R5-D4            97    32 <NA>    white,~ red          NA none   mascu~ Tatooi~
##  9 Biggs Dark~     183    84 black   light   brown        24 male   mascu~ Tatooi~
## 10 Obi-Wan Ke~     182    77 auburn~ fair    blue-g~      57 male   mascu~ Stewjon
## # ... with 77 more rows, 4 more variables: species <chr>, films <list>,
## #   vehicles <list>, starships <list>, and abbreviated variable names
## #   1: hair_color, 2: skin_color, 3: eye_color, 4: birth_year, 5: homeworld
```

```
starwars %>%
  filter (sex %in% c('male', 'female')) %>%
  ggplot(aes(x=sex, y=mass)) + geom_violin()
```

## Warning: Removed 23 rows containing non-finite values (`stat_ydensity()`).
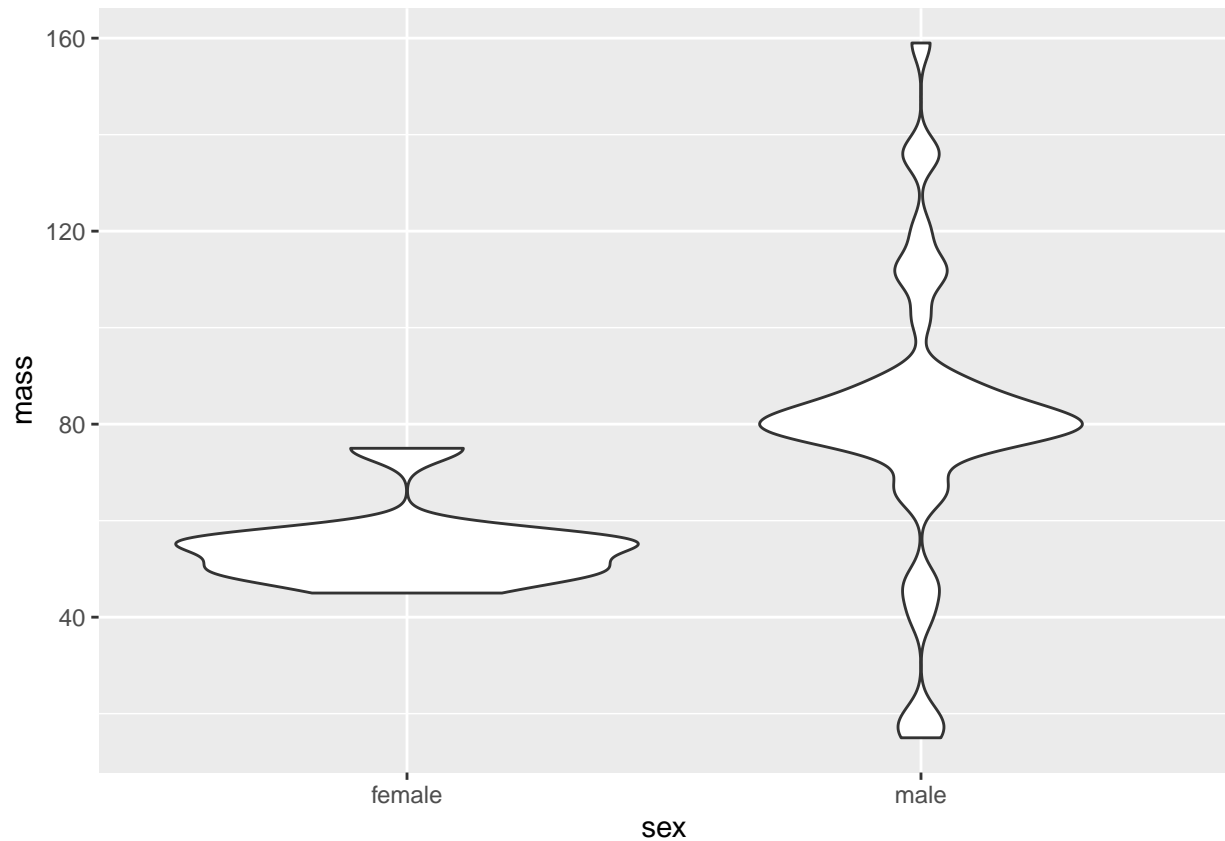


```
starwars %>%
  filter (sex %in% c('male', 'female')) %>%
  ggplot(aes(x=sex, y=mass, fill=sex)) +
  geom_violin(adjust=0.7) + # default adjust is 1. Lower means finer resolution
  guides(fill=FALSE) # make the entries coloured but delete the legend
```

## Warning: The `<scale>` argument of `guides()` cannot be `FALSE`. Use "none" instead as
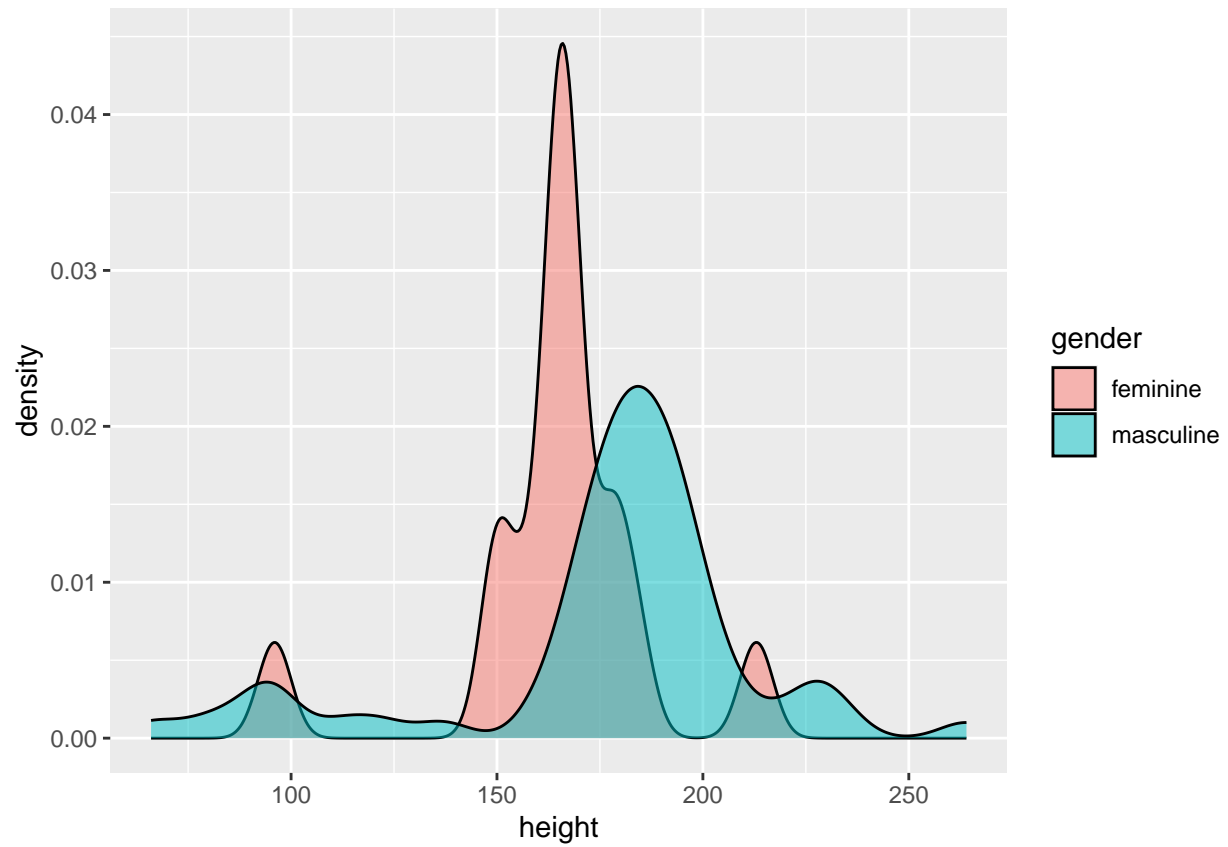## of ggplot2 3.3.4.
## Removed 23 rows containing non-finite values (`stat_ydensity()`).
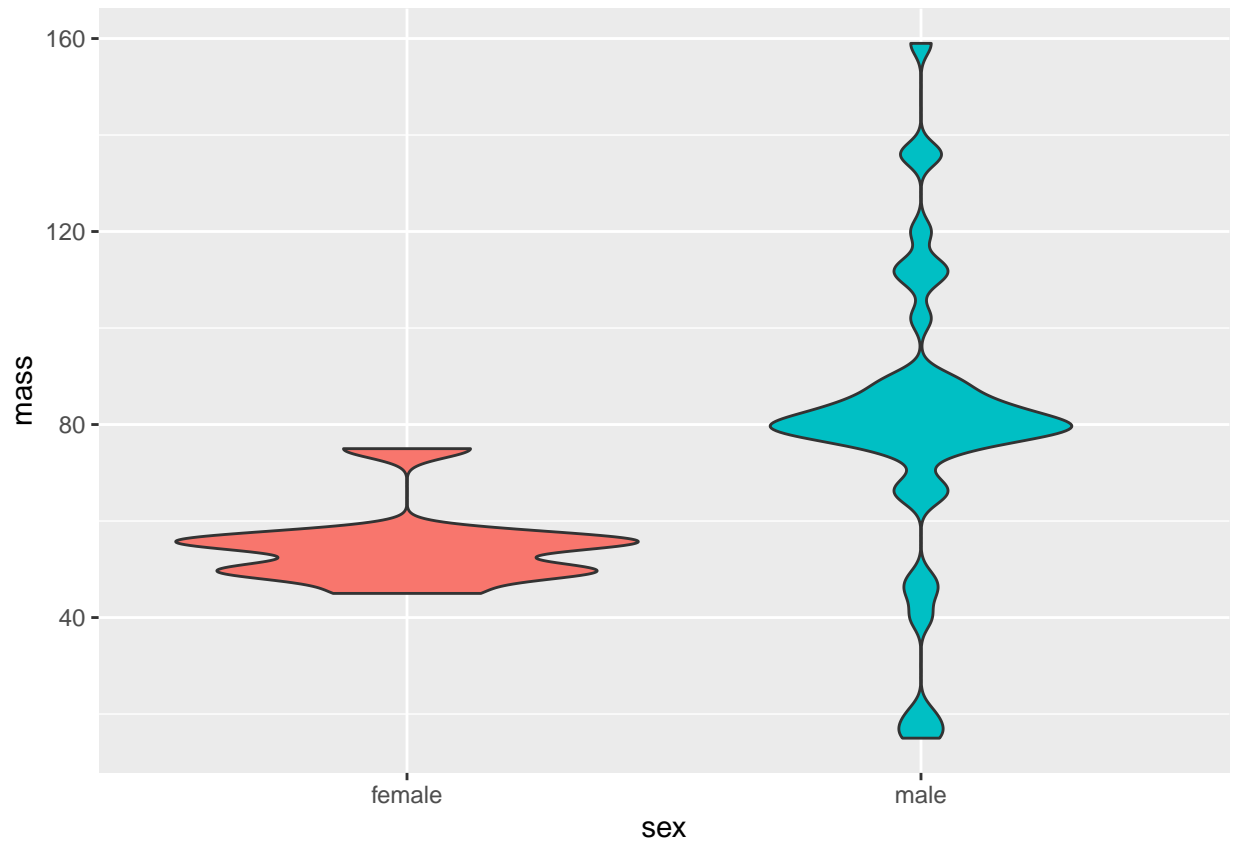
## Density plots

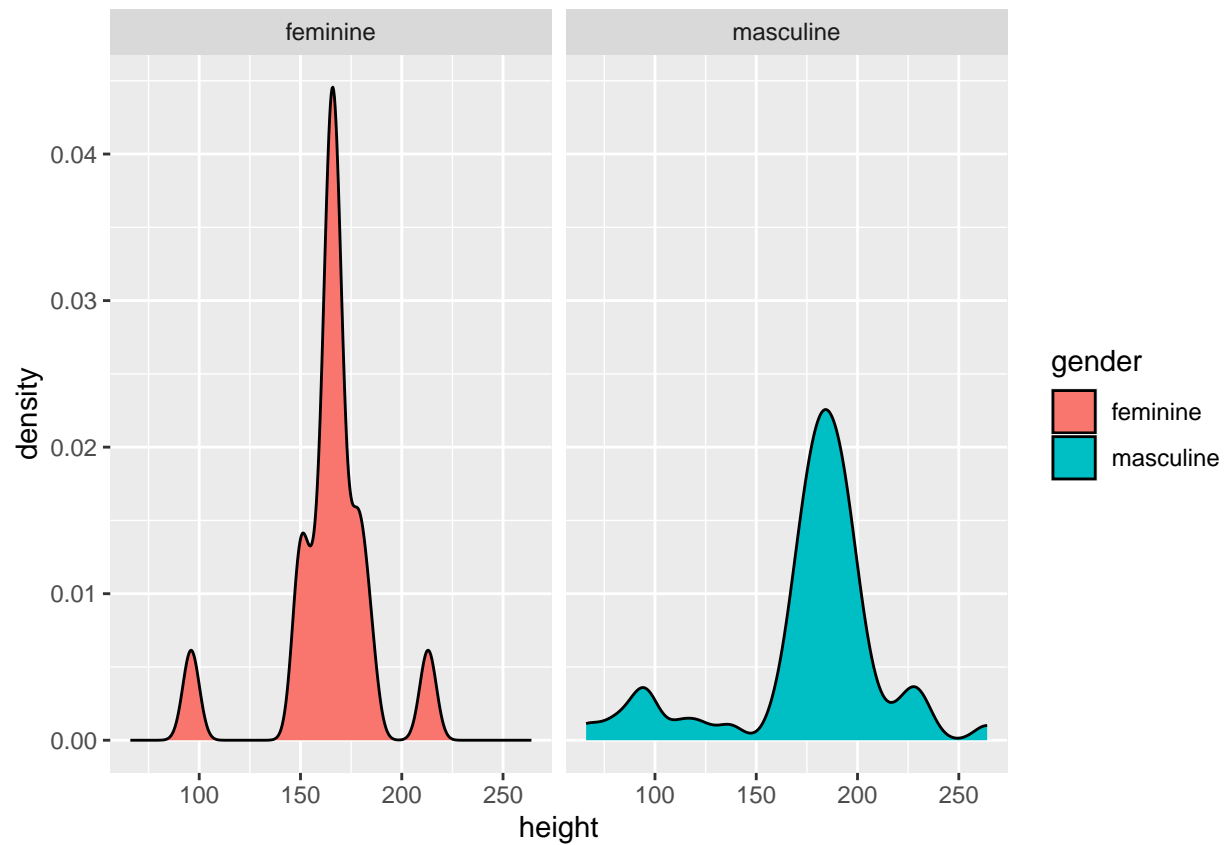They are like histograms, but with probability instead od count

```
starwars %>%
    filter(gender != "NA") %>%
    ggplot(aes(x = height, fill = gender)) + geom_density(alpha = 0.5)
```
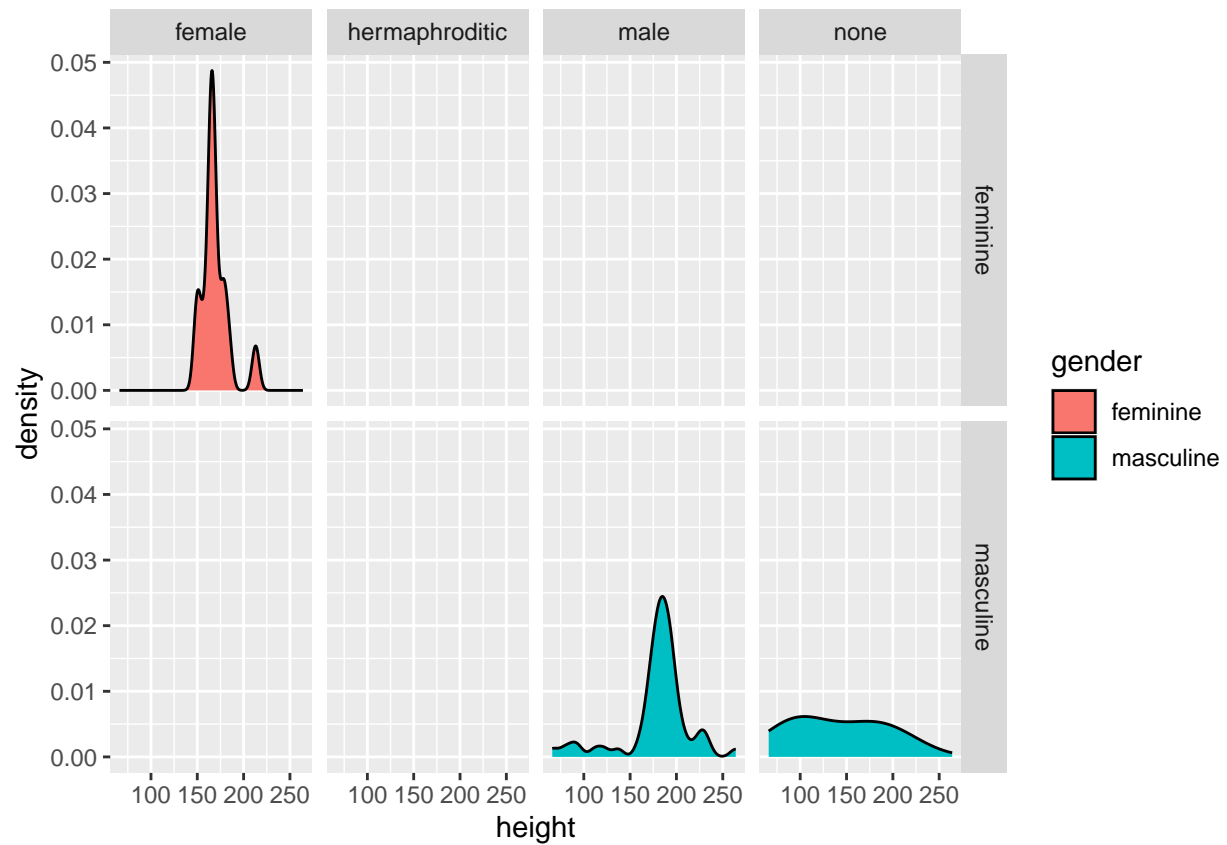
```
starwars %>%
    filter(gender != "NA") %>%
    ggplot(aes(x = height, fill = gender)) + geom_density() +
    facet_grid(gender ~ .)
```
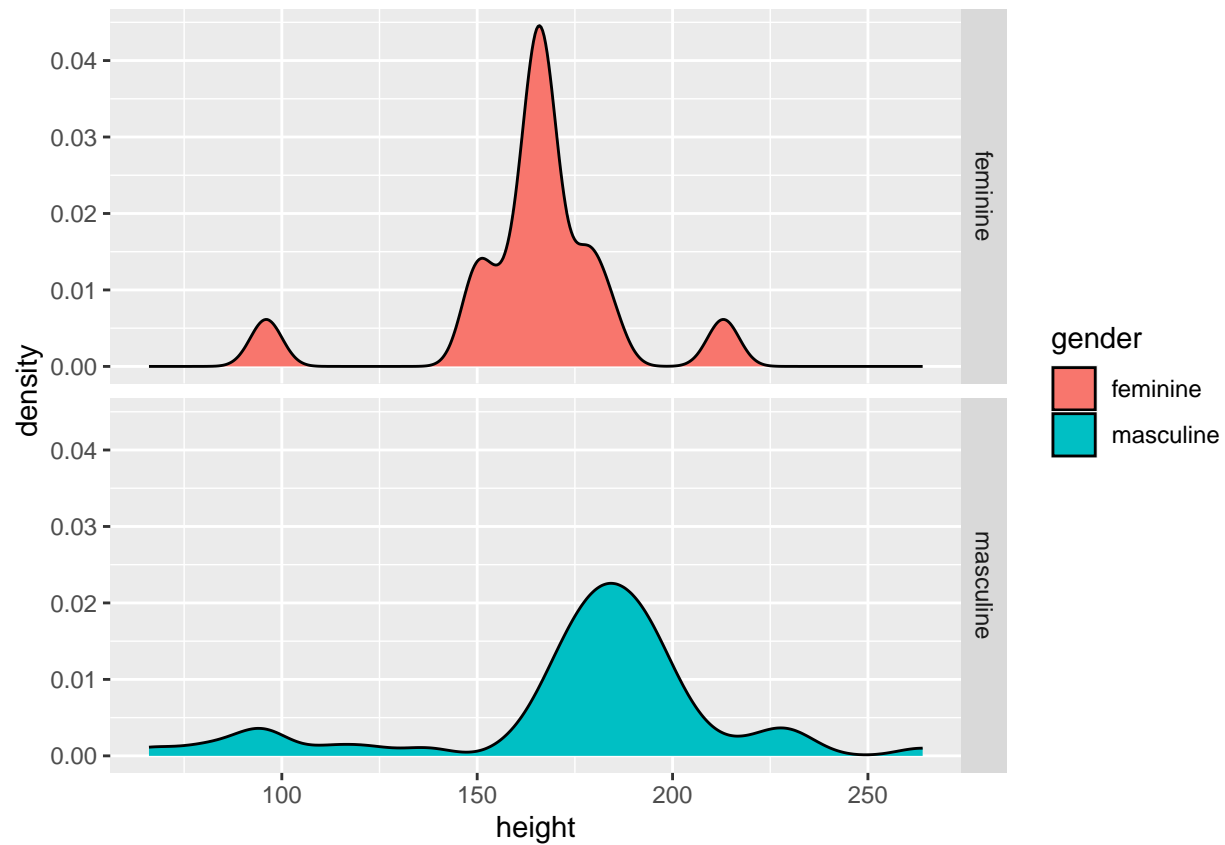
```
starwars %>%
    filter(gender != "NA") %>%
    ggplot(aes(x = height, fill = gender)) + geom_density() +
    facet_grid(. ~ gender)
```

```
starwars %>%
    filter(gender != "NA") %>%
    ggplot(aes(x = height, fill = gender)) + geom_density() +
    facet_grid(gender ~ sex)
```
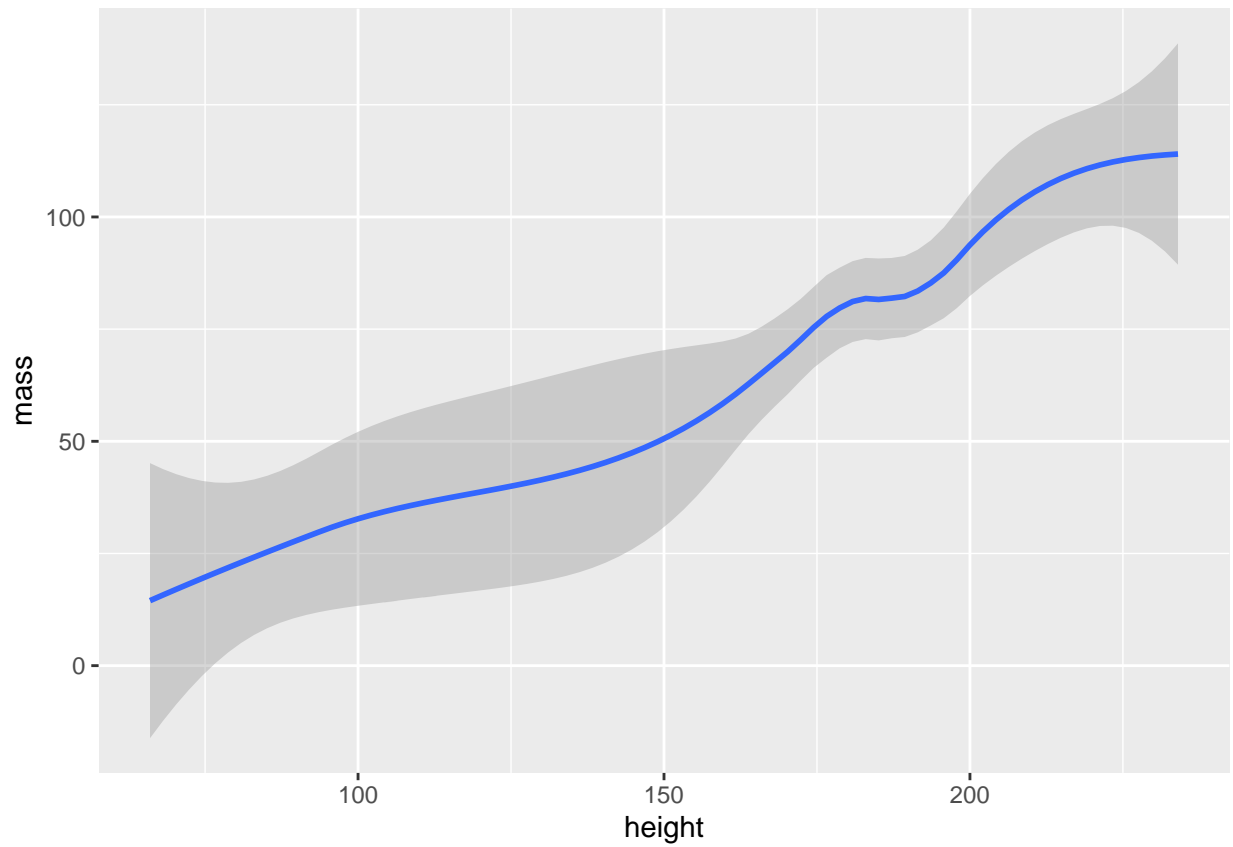
## Line graphs

```
starwars2 <- starwars %>%
    filter(sex %in% c("male", "female"), gender %in% c("masculine",
        "feminine"))


ggplot(starwars2, aes(x = height, y = mass)) + geom_smooth()
```
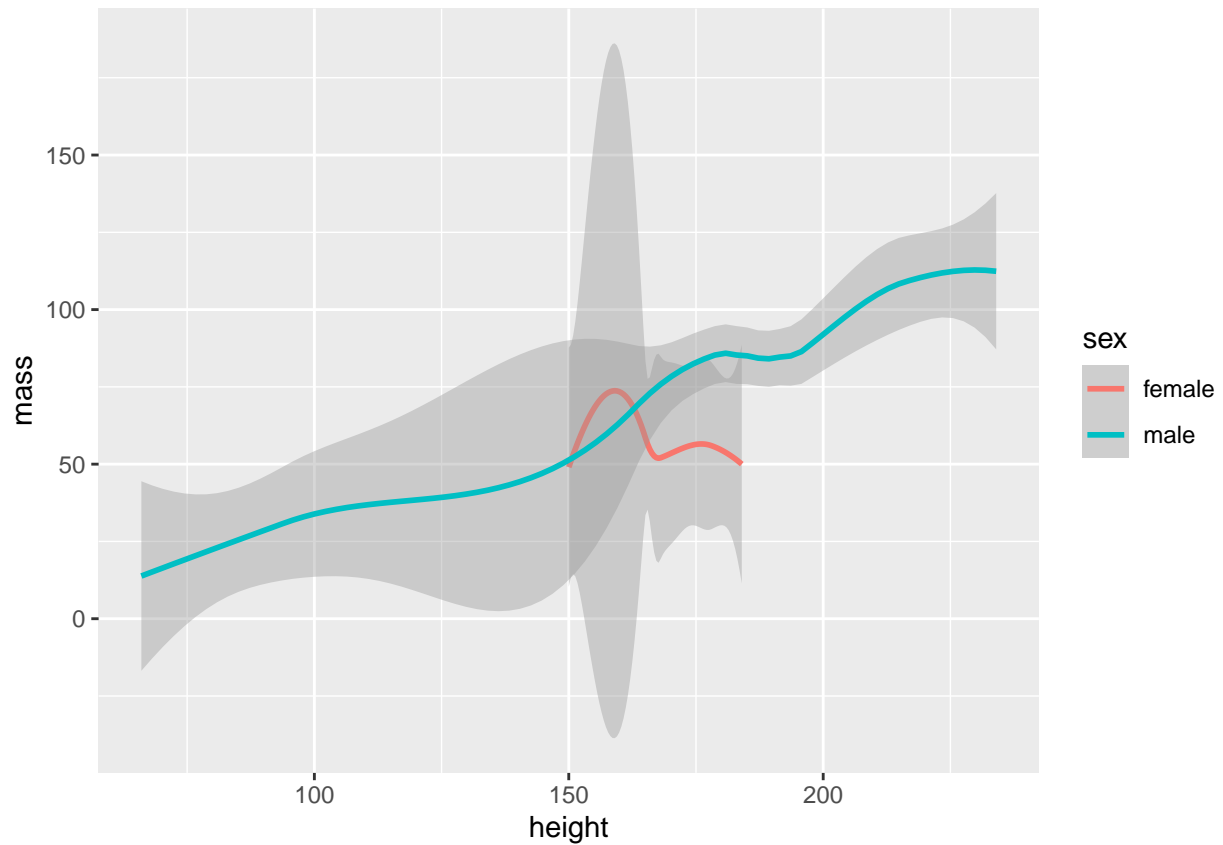
```
## `geom_smooth()` using method = 'loess' and formula = 'y ~ x'
```

```
ggplot(starwars2, aes(x = height, y = mass, color = sex)) + geom_smooth()
```

```
## 'geom_smooth()' using method = 'loess' and formula = 'y ~ x'
```

```
ggplot(starwars2, aes(x = height, y = mass, colour = sex)) +
    geom_line(size = 3) + theme(axis.line = element_line(size = 0.5),
    panel.background = element_rect(fill = NA, size = rel(20)),
    panel.grid.minor = element_line(colour = NA), axis.text = element_text(size = 16),
    axis.title = element_text(size = 18)) + scale_y_continuous(expand = c(0,
    0))
```
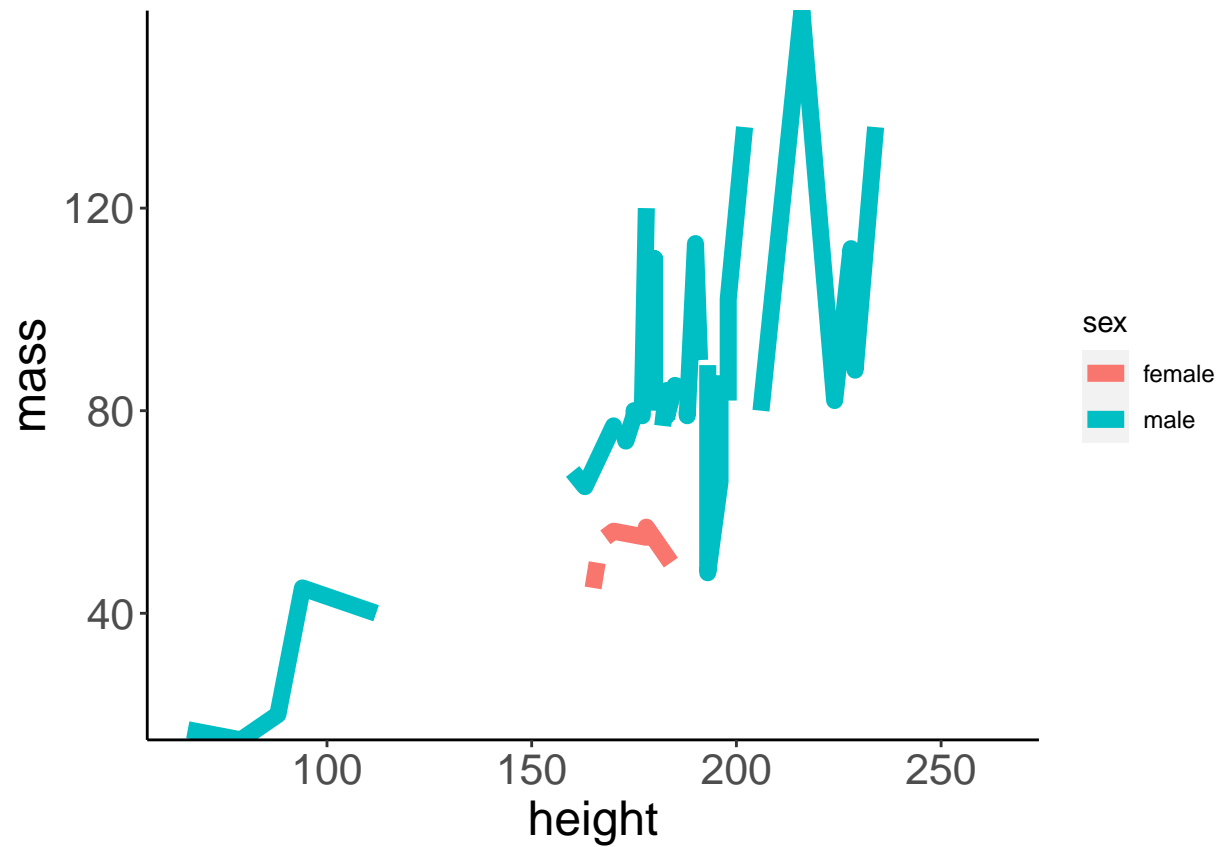
Orange

```
## Grouped Data: circumference ~ age | Tree
##    Tree  age circumference
## 1     1  118            30
## 2     1  484            58
## 3     1  664            87
## 4     1 1004           115
## 5     1 1231           120
## 6     1 1372           142
## 7     1 1582           145
## 8     2  118            33
## 9     2  484            69
## 10    2  664           111
## 11    2 1004           156
## 12    2 1231           172
## 13    2 1372           203
## 14    2 1582           203
## 15    3  118            30
## 16    3  484            51
## 17    3  664            75
## 18    3 1004           108
## 19    3 1231           115
## 20    3 1372           139
## 21    3 1582           140
## 22    4  118            32
```
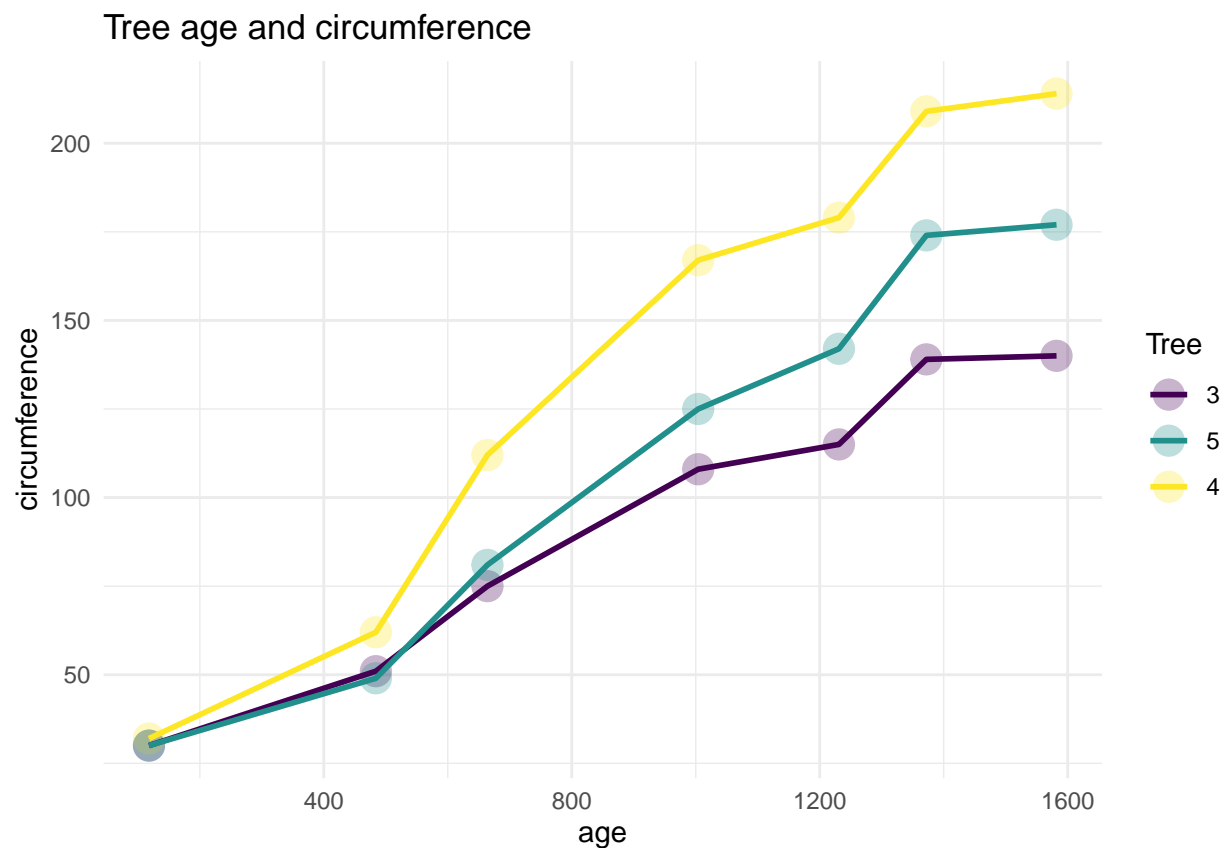
```
## 23   4  484          62
## 24   4  664         112
## 25   4 1004         167
## 26   4 1231         179
## 27   4 1372         209
## 28   4 1582         214
## 29   5  118          30
## 30   5  484          49
## 31   5  664          81
## 32   5 1004         125
## 33   5 1231         142
## 34   5 1372         174
## 35   5 1582         177
```

```
Orange %>%
    filter(Tree != "1" & Tree != "2") %>%
    ggplot(aes(x = age, y = circumference, colour = Tree)) +
    geom_point(size = 5, alpha = 0.3) + geom_line(size = 1) +
    theme_minimal() + labs(title = "Tree age and circumference")
```



Tree age and circumference

```
# line graphs with error bars (SEM) + customly-coloured
# SEM, mean, custom colours
library(ggplot2)

sem <- function(x, na.rm = FALSE) {
```

```
    out <- sd(x, na.rm = na.rm)/sqrt(length(x))
    return(out)
}
pointcolour <- c(Fair = "yellow", Good = "red", `Very Good` = "pink",
    Premium = "blue", Ideal = "black")
# write.csv(diamonds, file='diamonds.csv', row.names=TRUE)
```
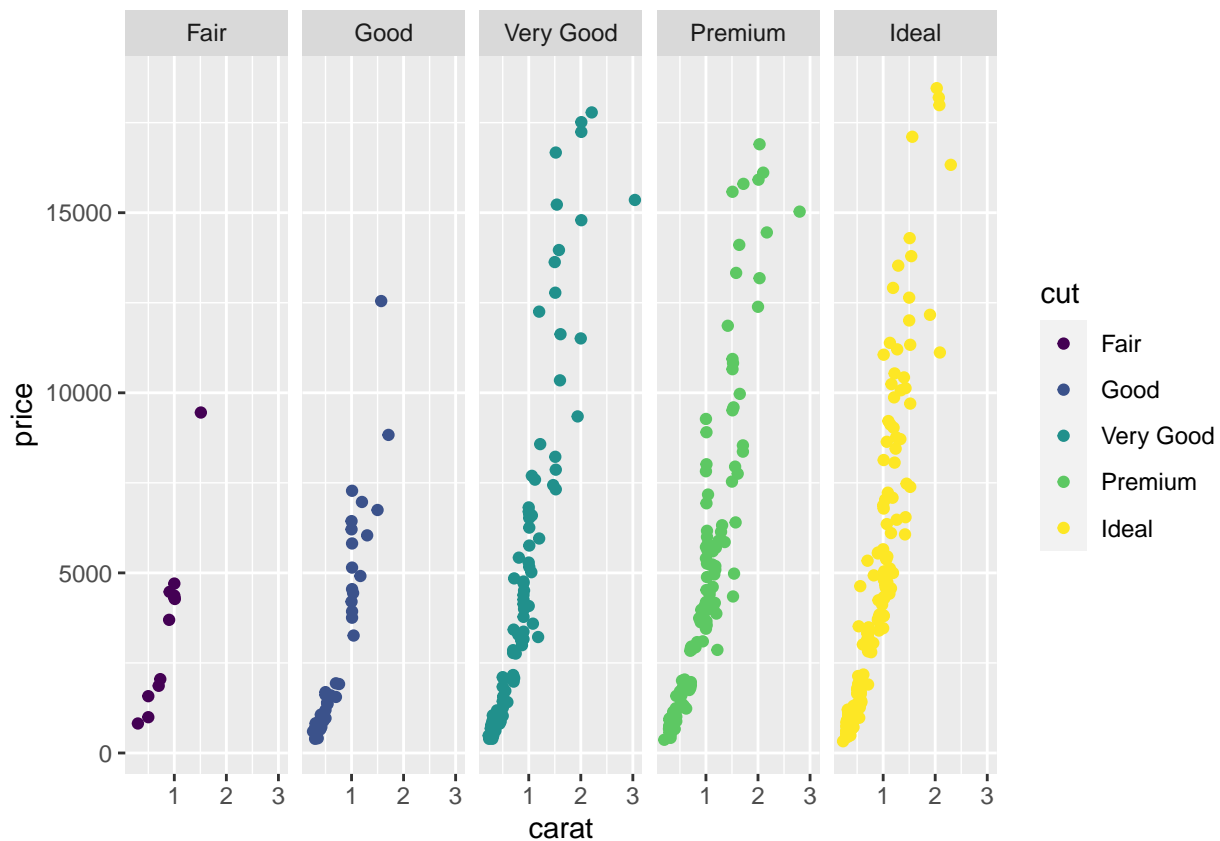
## Pie charts

Nothing here right now. . .

## Tiered faceted graphs

```
library(ggpubr)

p1 <- ggplot(diamonds[seq(1, 50000, by = 100),], aes(x = carat, y = price, colour = cut)) + geom_point(
p1
```
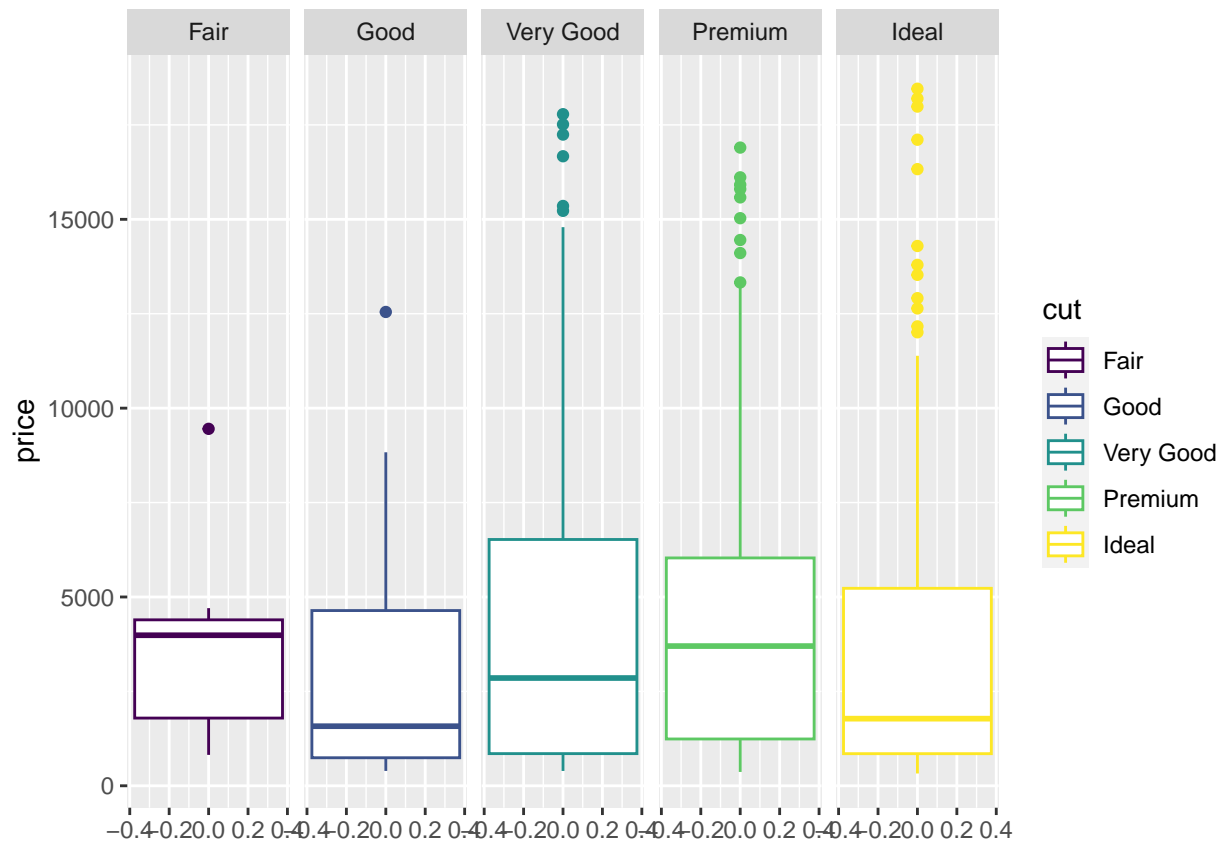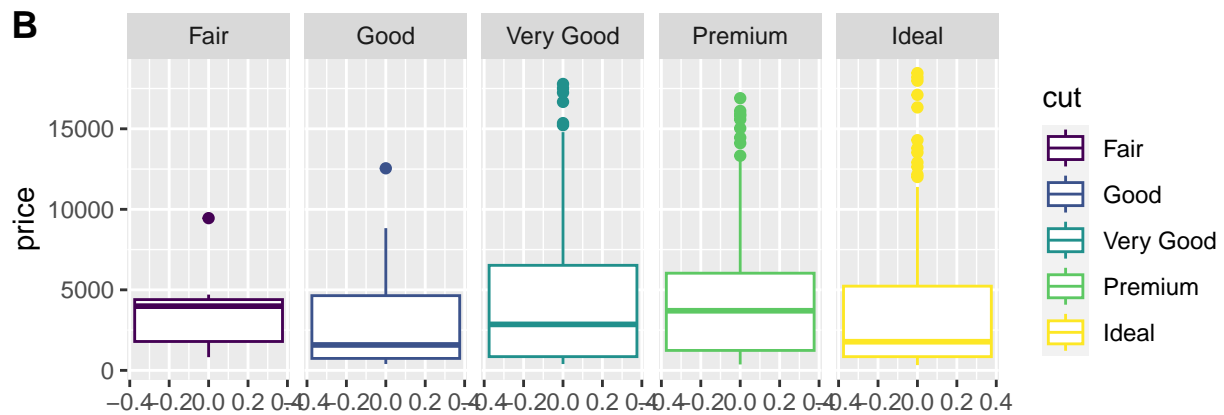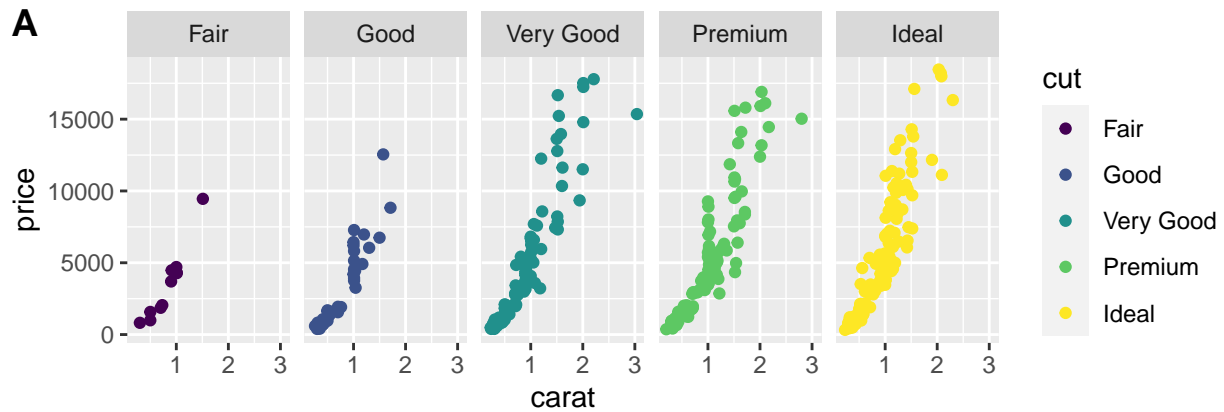


```
p2 <- ggplot(diamonds[seq(1, 50000, by = 100),], aes(y = price, colour = cut)) + geom_boxplot() + facet_
p2
```
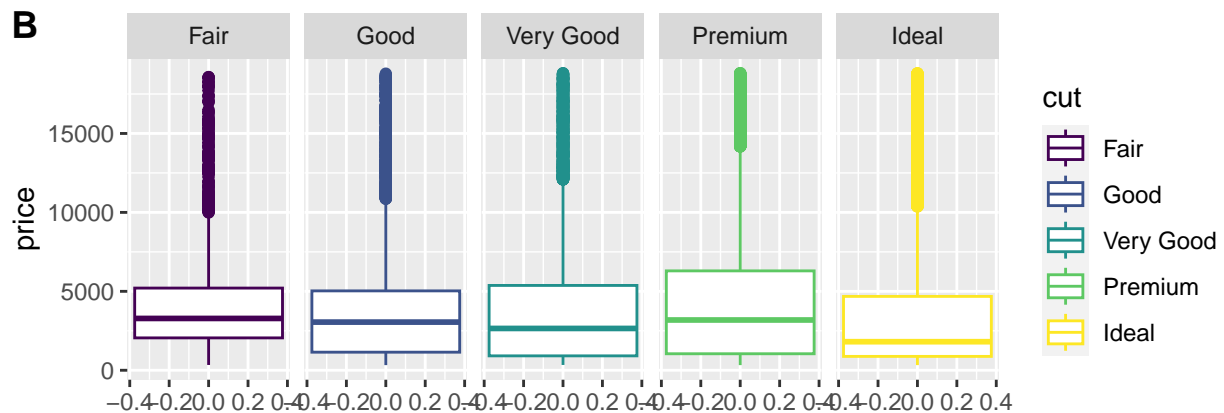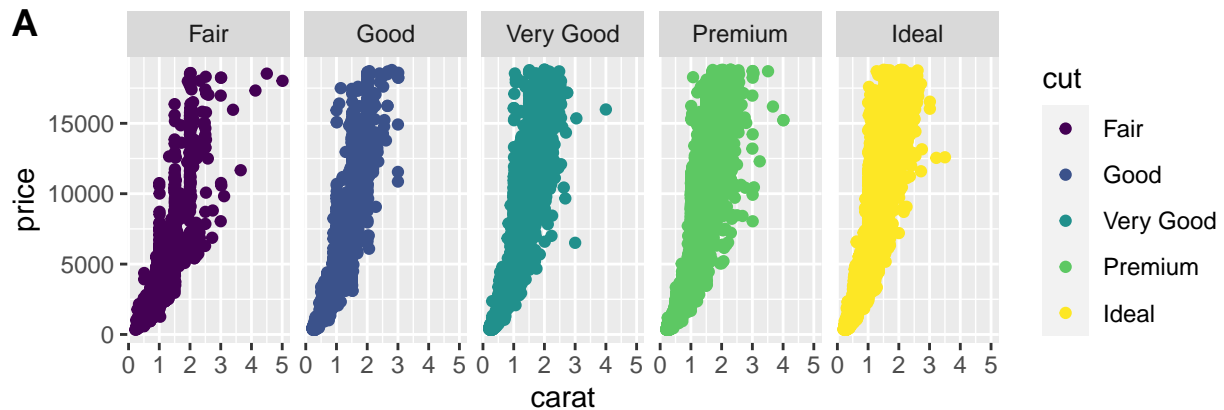
```
ggarrange(p1, p2, nrow = 2, labels = "AUTO")
```

```
# OR the same, just the entire dataset & without creating new variables #
ggarrange(
  # graph 1
  ggplot(diamonds, aes(x = carat, y = price, colour = cut)) + geom_point() + facet_wrap( ~ cut, nrow = 
  # graph 2
  ggplot(diamonds, aes(y = price, colour = cut)) + geom_boxplot() + facet_wrap( ~ cut, nrow = 1),
  # ggarrange settings
  nrow = 2, labels = "AUTO"
)
```

```
summary(relig_income)
```

religion <$10k $10-20k $20-30k
Length:18 Min. : 1.00 Min. : 2.00 Min. : 3.0
Class :character 1st Qu.: 12.25 1st Qu.: 14.75 1st Qu.: 17.0
Mode :character Median : 20.00 Median : 27.00 Median : 33.5
Mean :107.22 Mean :154.50 Mean : 186.5
3rd Qu.:170.00 3rd Qu.:193.00 3rd Qu.: 192.0
Max. :575.00 Max. :869.00 Max. :1064.0
$30-40k $40-50k $50-75k $75-100k
Min. : 4.00 Min. : 2.0 Min. : 7.00 Min. : 3.00
1st Qu.: 15.75 1st Qu.: 15.0 1st Qu.: 34.25 1st Qu.: 25.25
Median : 40.00 Median : 34.0 Median : 66.50 Median : 65.50
Mean :183.44 Mean :171.4 Mean : 288.06 Mean :221.67
3rd Qu.:198.75 3rd Qu.:166.8 3rd Qu.: 201.50 3rd Qu.:128.75
Max. :982.00 Max. :881.0 Max. :1486.00 Max. :949.00
$100-150k >150k Don't know/refused Min. : 4.0 Min. : 4.00 Min. : 8.00
1st Qu.: 22.5 1st Qu.: 23.75 1st Qu.: 41.25
Median : 48.5 Median : 53.50 Median : 74.50
Mean :177.6 Mean :144.89 Mean : 340.06
3rd Qu.:103.5 3rd Qu.:134.25 3rd Qu.: 294.75
Max. :792.0 Max. :634.00 Max. :1529.00

```
p1 <- relig_income %>%
  filter(religion == "Atheist" |
```

```r
        religion == "Mormon" |
        religion == "Buddhist") %>%
  melt(id.vars = "religion") %>%
  group_by(religion, variable) %>%
  ggplot(aes(x = religion, y = value, fill = variable)) +
  geom_bar(stat="identity", position = "fill") +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        axis.title.x = element_blank(),
        panel.background = element_blank(),
        axis.line = element_line(size = 0.5)) +
  scale_color_brewer(palette = "Set1") +
  labs(title = "Distribution of income levels by religion",
        y = "Percentage")
p2 <- relig_income %>%
  filter(religion == "Atheist" |
        religion == "Mormon" |
        religion == "Buddhist") %>%
  melt(id.vars = "religion") %>%
  group_by(religion, variable) %>%
  ggplot(aes(x = religion, y = value)) +
  geom_bar(stat="identity") +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        axis.title = element_text(),
        panel.background = element_blank(),
        axis.line = element_line(size = 0.5)) +
  scale_color_brewer(palette = "Set1") +
  labs(title = "Total count of observations by religion",
        x = "Religion",
        y = "Count")
ggarrange(p1, p2, nrow = 2, labels = "AUTO")
```

**A**  Distribution of income levels by religion



**B**  Total count of observations by religion
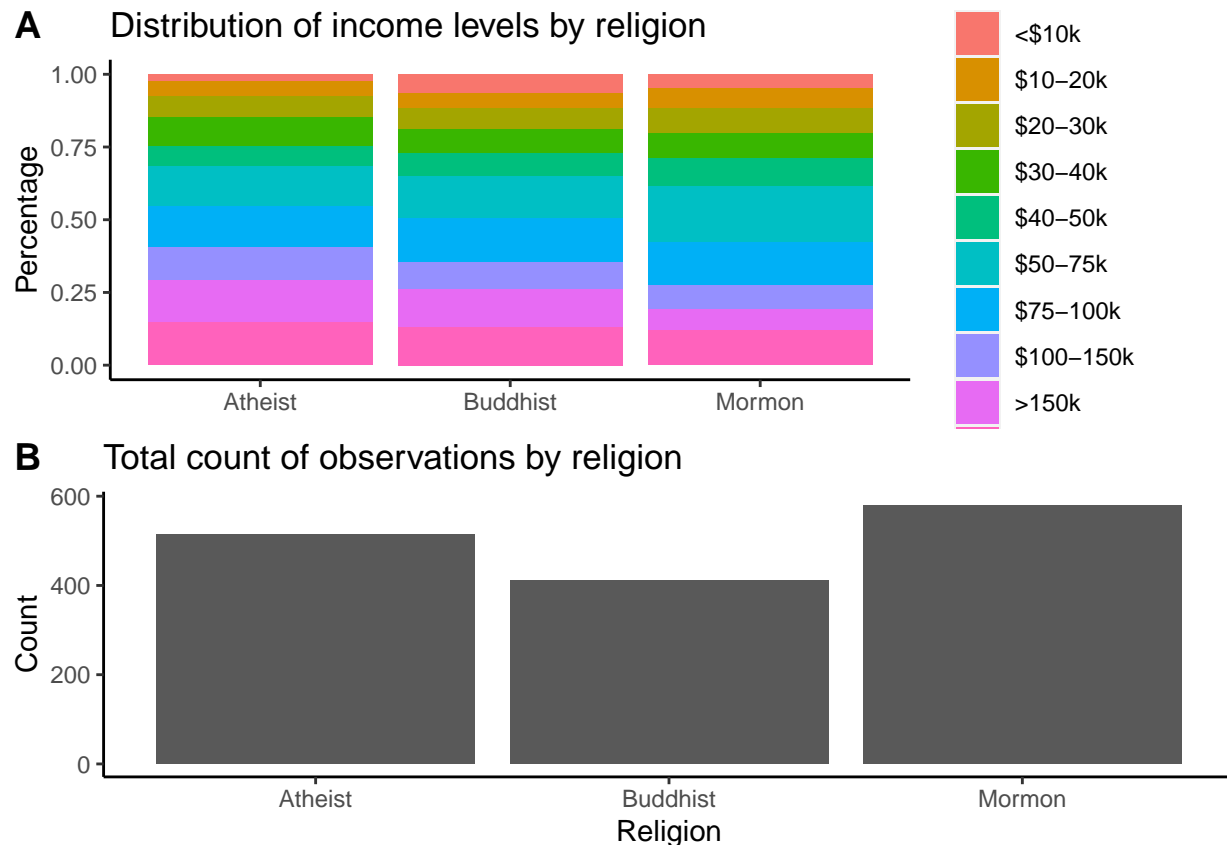


# Graph customisation

## Theme settings

- Flip coordinates: + coord_flip()
- Log scale: + scale_y_log10()
- Show fonts: fonts()
- Globally change theme settings: theme_set(theme_gray(base_size=20))

## Colour changes

- Globally reset theme settings to default: theme_set(theme_gray())
- Black and white theme: mainplot + theme_bw()
- Manually assign colours: scale_color_manual(values = c("#A6611A", "#DFC27D", "#6e6c6b", "#80CDC1", "#018571"))

## Colour palettes

- library(RColorBrewer)
- display.brewer.all()
- colours() # gives names of different colours
- mainplot + scale_fill_brewer(palette="Set1")

- Setting colour values for each of the 5 variables in my mainplot graph manually: mainplot + scale_fill_manual(values = c("green", "blue", "red", "orange", "grey"))

## Changing theme

+ `theme()` argument

- Number size on axis: axis.text = element_text(size = 20)
- Title size on axis: axis.title = element_text(size = 20)
- Legend: legend.title = element_text(size = 20)
- Background: panel.background = element_rect(fill = "pink")
- Change major gridlines: panel.grid.major = element_line(colour = "blue")

  - size = 0.2
  - NA

- Change minor gridlines: panel.grid.minor = element_line(colour = "red")
- Change tick marks: axis.ticks = element_line(size=2, colour="blue")

  - As options: axis.ticks.x, axis.ticks.y

- Change legend position: legend.position="top"

  - Other arguments:
  - "bottom"
  - Bottom left: c(0,0)
  - Near the top right: c(0.8,0.8)

## Changing labels

+ `labs()` argument

- title = "Title"
- y = "Title y"
- x = "Title x"

## Publication style

Clear background, axis lines, no box, no grid lines, basic colors, no legend

```
# Just for one plot
mainplot + theme(axis.line = element_line(size = 0.5), panel.background = element_rect(fill = NA,
    size = rel(20)), panel.grid.minor = element_line(colour = NA),
    axis.text = element_text(size = 16), axis.title = element_text(size = 18)) +
    scale_y_continuous(expand = c(0, 0))  # to stop the bars from floaing above the x-axis

# Globally apply this theme
theme_set(theme_gray() + theme(axis.line = element_line(size = 0.5),
    panel.background = element_rect(fill = NA, size = rel(20)),
    panel.grid.minor = element_line(colour = NA), axis.text = element_text(size = 16),
    axis.title = element_text(size = 18)))
```