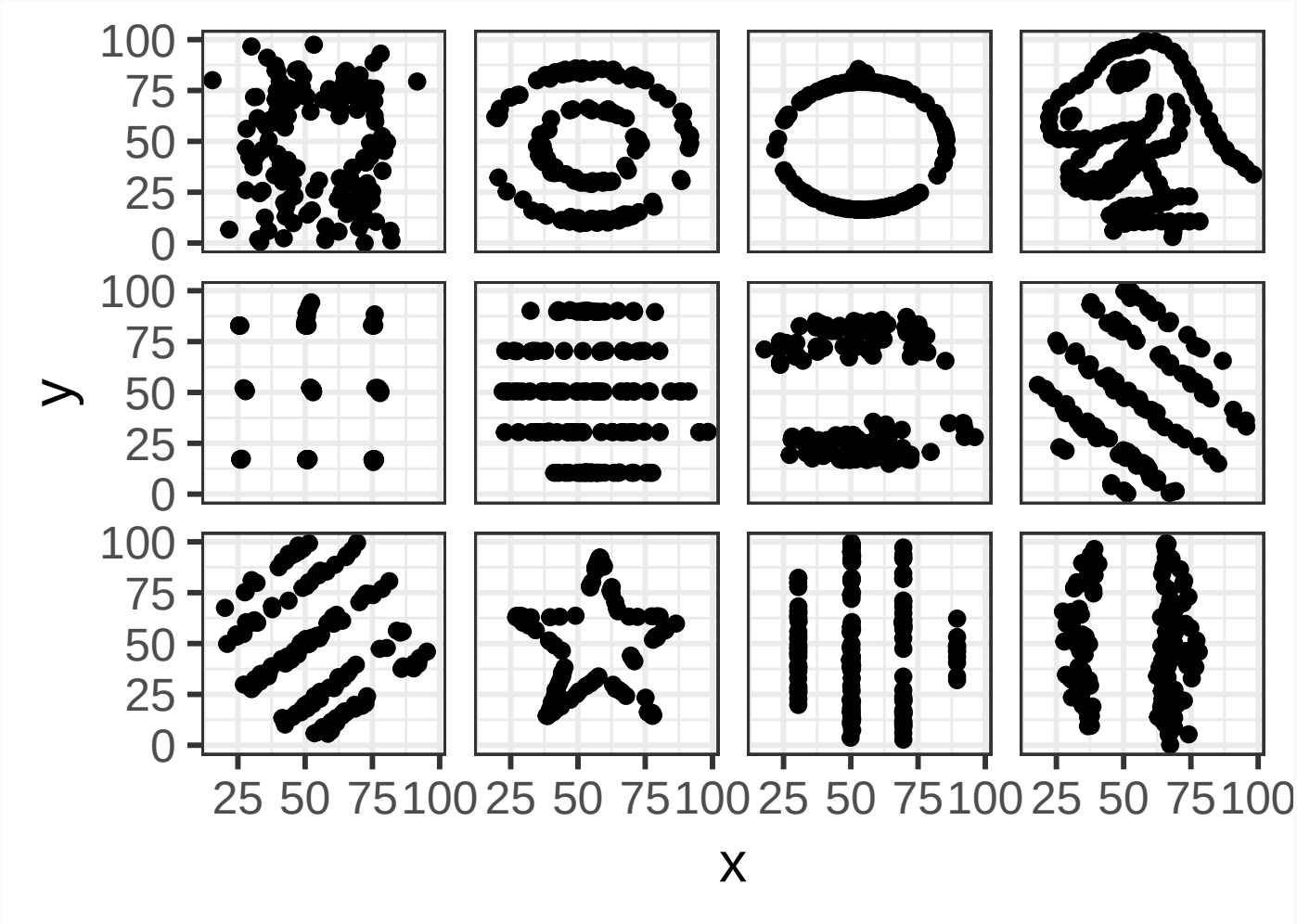
# Data visualisation with ggplot2

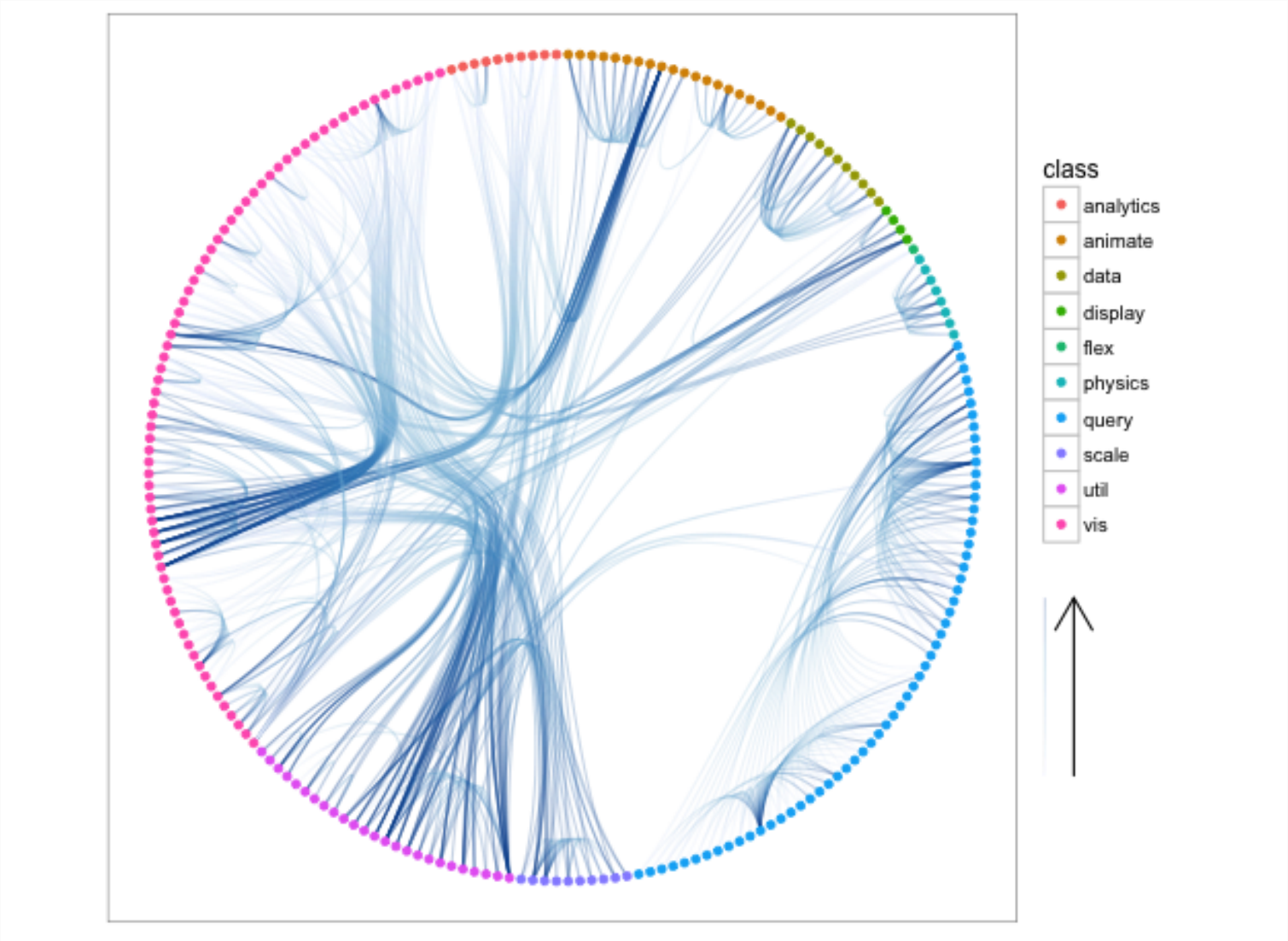
Francisco Rodriguez-Sanchez (@frod_san)

# Always plot data!




https://github.com/stephlocke/datasauRus

# Made with ggplot

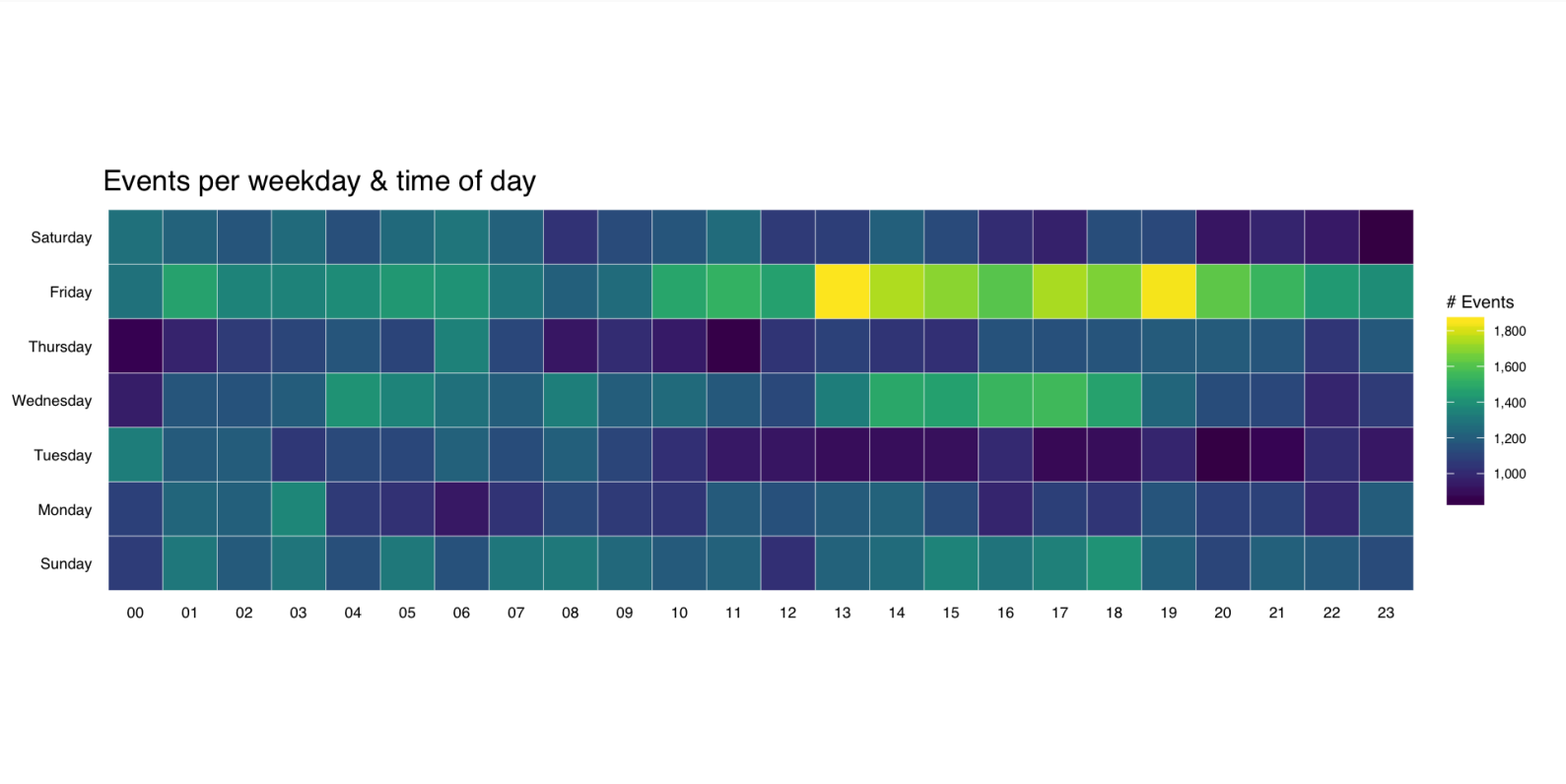



https://github.com/thomasp85/ggraph

# Made with ggplot

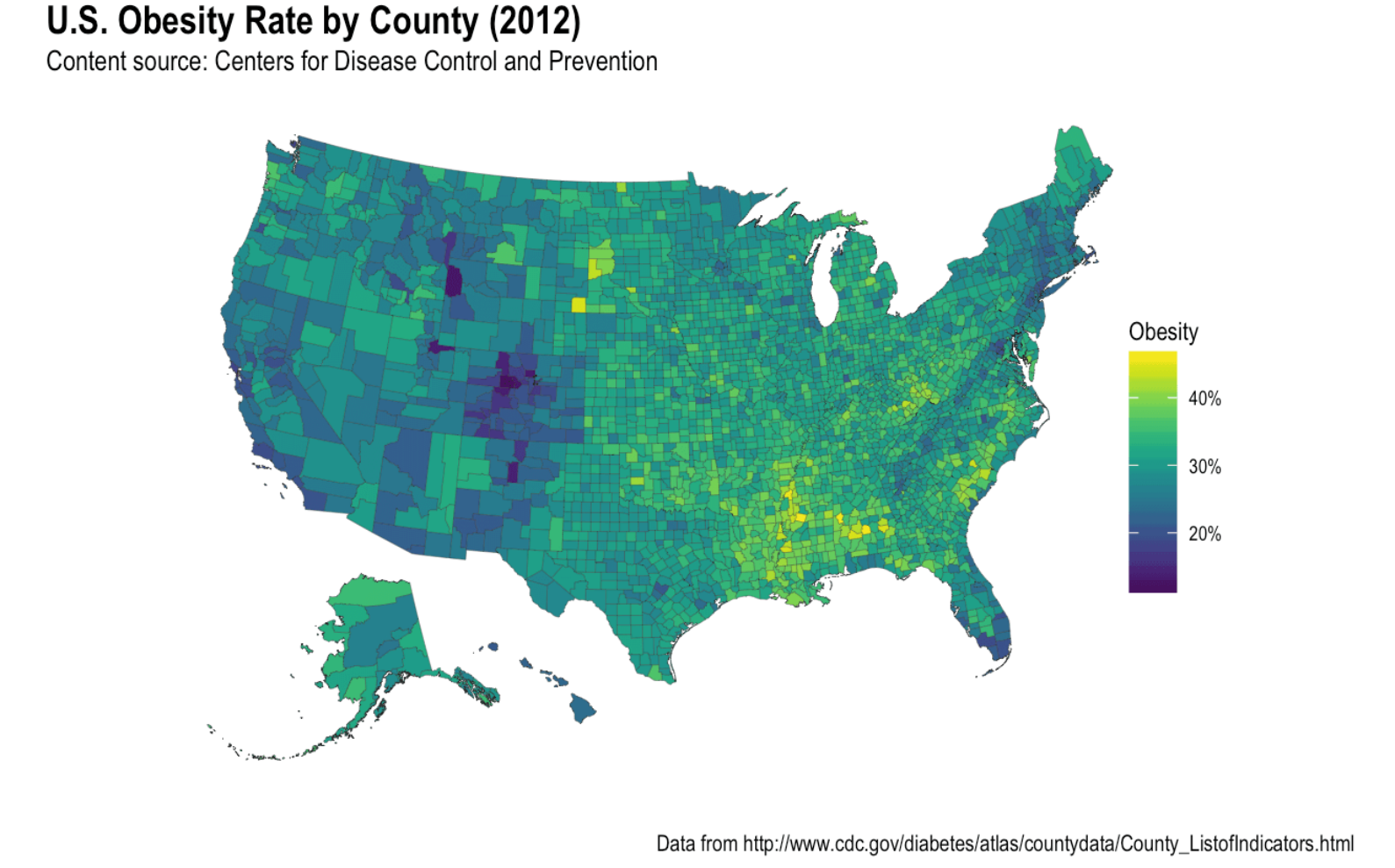

http://spatial.ly/2012/02/great-maps-ggplot2/

# Made with ggplot




https://rud.is/b/2016/02/14/making-faceted-heatmaps-with-ggplot2/

# Made with ggplot

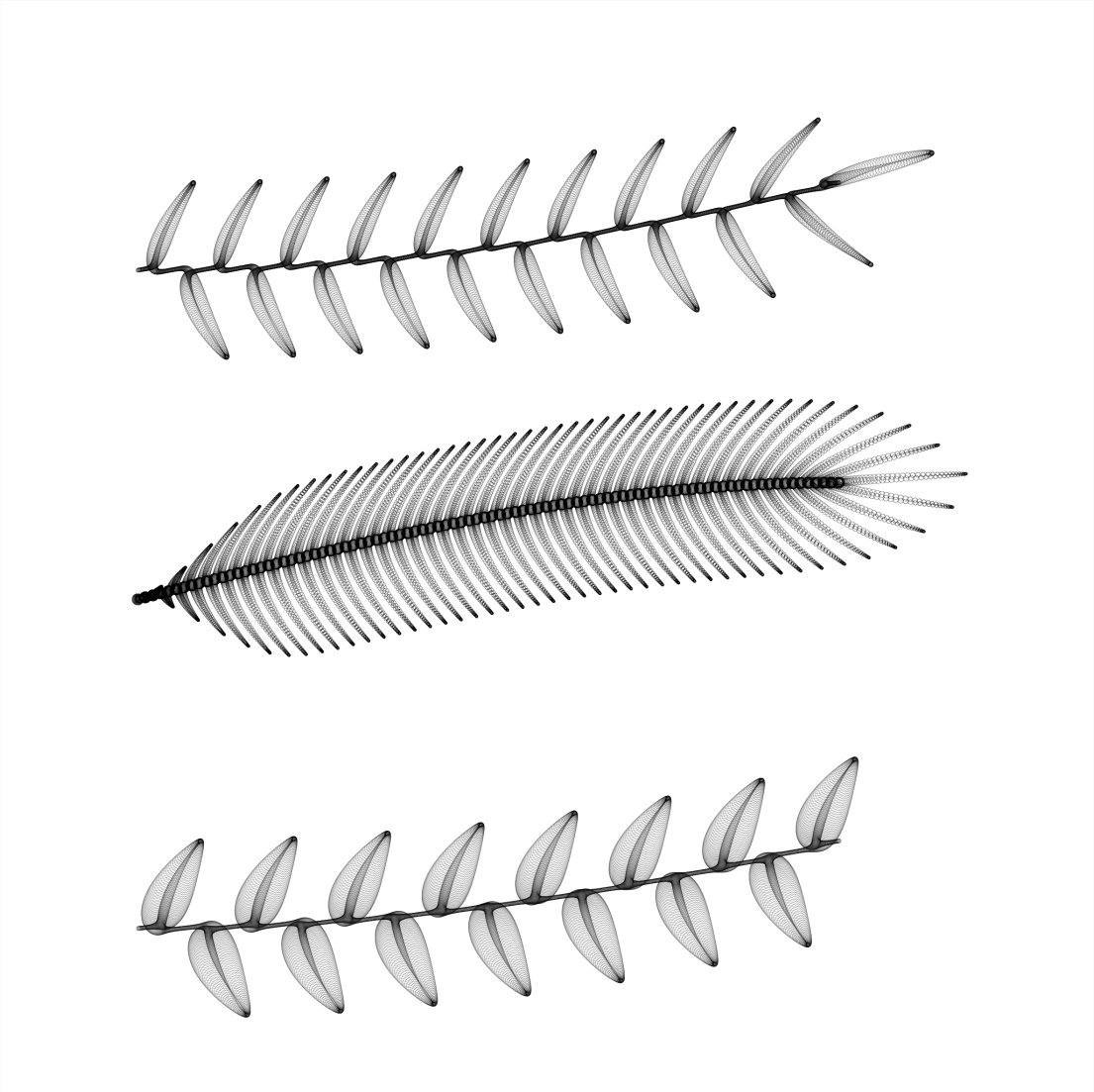



https://rud.is/b/2016/03/29/easier-composite-u-s-choropleths-with-albersusa/

# Made with ggplot


https://guangchuangyu.github.io/ggtree/

# Made with ggplot



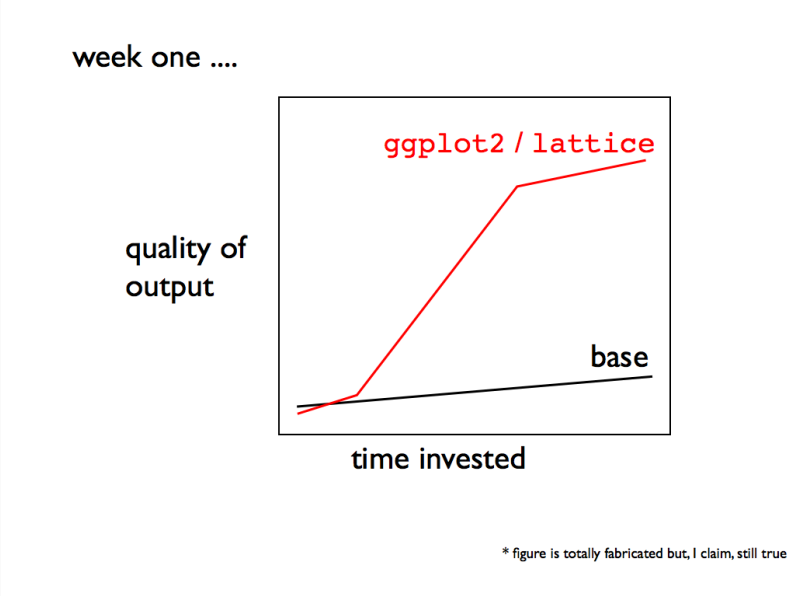
https://github.com/marcusvolz/mathart

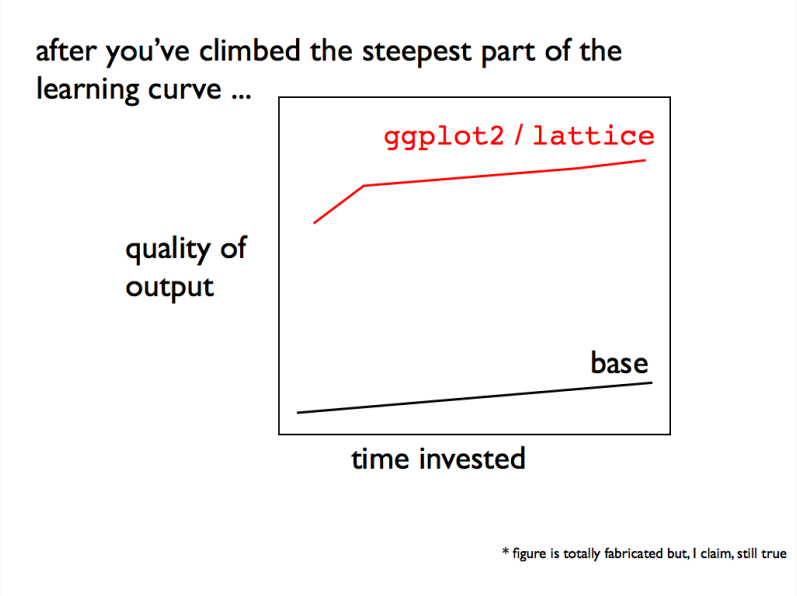
# Why ggplot?

# Why ggplot

- Extremely powerful and flexible
- Consistent (grammar of graphics)
- Very powerful user base and active development

# At the beginnning it's hard, but then it pays off



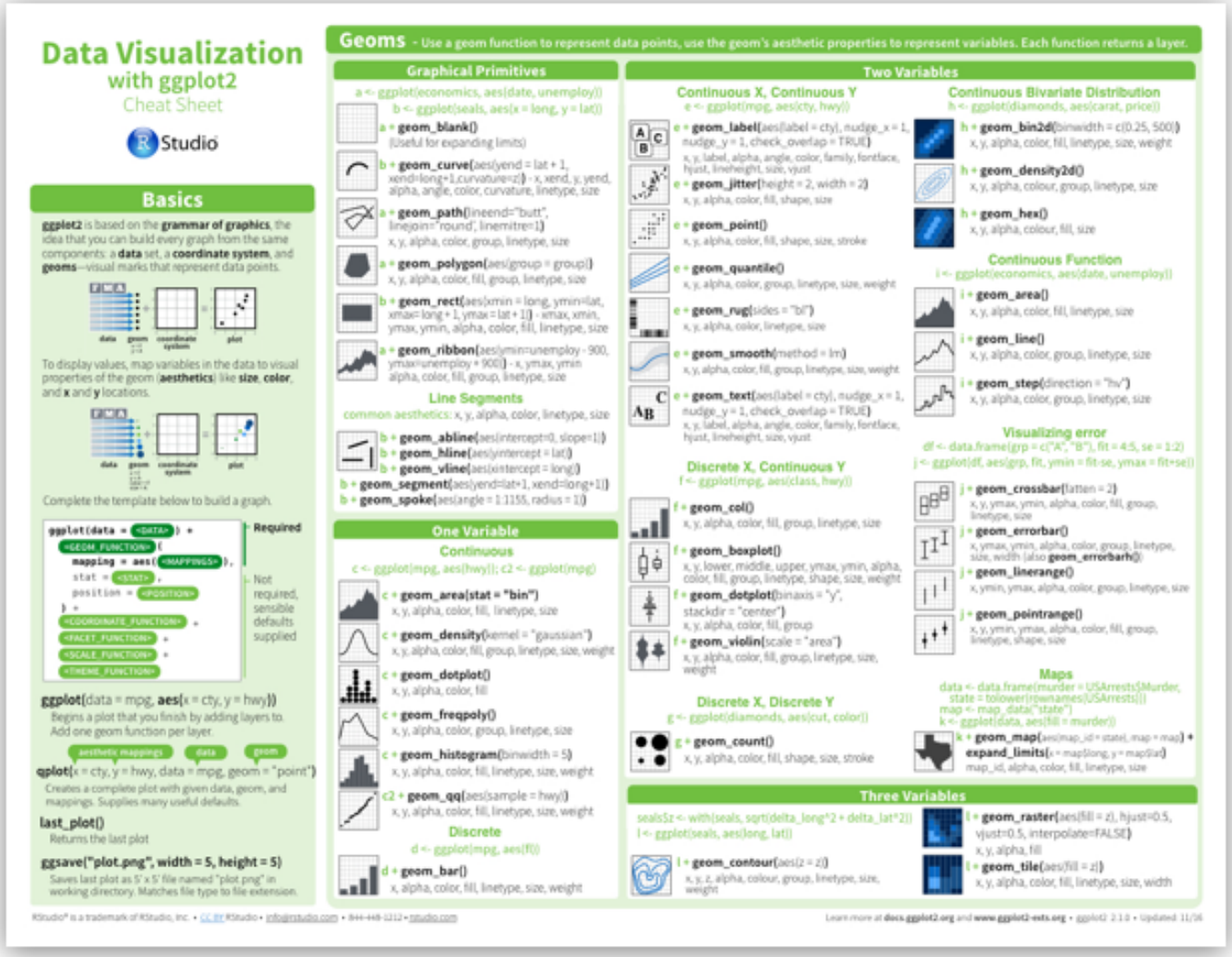




Source: https://github.com/jennybc/ggplot2-tutorial

# Very good documentation and tutorials

- Official ggplot2 documentation
- ggplot2 book
- R graphics cookbook and Cookbook for R
- Beautiful plotting in R: A ggplot2 cheatsheet
- Introduction to ggplot2
- Tutorial: ggplot2
- How to format plots for publication using ggplot2
- Visualising data with ggplot2
- Data Visualization with R and ggplot2
- ggplot2 tutorial
- Data visualisation chapter in R for Data Science
- The complete ggplot2 tutorial
- Data visualization: a practical introduction (K. Healy)
- Fundamentals of data visualization (C. Wilke)

# Cheatsheet



Data Visualization with ggplot2 Cheat Sheet

RStudio

Basics

ggplot2 is based on the grammar of graphics, the idea that you can build every graph from the same components: a data set, a coordinate system, and geoms—visual marks that represent data points.

To display values, map variables in the data to visual properties of the geom (aesthetics) like size, color, and x and y locations.

Complete the template below to build a graph.

```
ggplot(data = <DATA>) +
  <GEOM_FUNCTION>(
    mapping = aes(<MAPPINGS>),
    stat = <STAT>,
    position = <POSITION>
  ) +
  <COORDINATE_FUNCTION> +
  <FACET_FUNCTION> +
  <SCALE_FUNCTION> +
  <THEME_FUNCTION>
```

Required

Not required, sensible defaults supplied

ggplot(data = mpg, aes(x = cty, y = hwy))
Begins a plot that you finish by adding layers to. Add one geom function per layer.

aesthetic mappings data geom

qplot(x = cty, y = hwy, data = mpg, geom = "point")
Creates a complete plot with given data, geom, and mappings. Supplies many useful defaults.

last_plot()
Returns the last plot

ggsave("plot.png", width = 5, height = 5)
Saves last plot as 5' x 5' file named "plot.png" in working directory. Matches file type to file extension.

Geoms - Use a geom function to represent data points, use the geom's aesthetic properties to represent variables. Each function returns a layer.

Graphical Primitives

a <- ggplot(economics, aes(date, unemploy))
b <- ggplot(seals, aes(x = long, y = lat))

a + geom_blank()
(Useful for expanding limits)

b + geom_curve(aes(yend = lat + 1, xend=long+1,curvature=z)) - x, xend, y, yend, alpha, angle, color, curvature, linetype, size

a + geom_path(lineend="butt", linejoin="round", linemitre=1)
x, y, alpha, color, group, linetype, size

a + geom_polygon(aes(group = group))
x, y, alpha, color, fill, group, linetype, size

b + geom_rect(aes(xmin = long, ymin=lat, xmax= long + 1, ymax = lat + 1)) - xmax, xmin, ymax, ymin, alpha, color, fill, linetype, size

a + geom_ribbon(aes(ymin=unemploy - 900, ymax=unemploy + 900)) - x, ymax, ymin
alpha, color, fill, group, linetype, size

Line Segments

common aesthetics: x, y, alpha, color, linetype, size

b + geom_abline(aes(intercept=0, slope=1))
b + geom_hline(aes(yintercept = lat))
b + geom_vline(aes(xintercept = long))
b + geom_segment(aes(yend=lat+1, xend=long+1))
b + geom_spoke(aes(angle = 1:1155, radius = 1))

One Variable

Continuous

c <- ggplot(mpg, aes(hwy)); c2 <- ggplot(mpg)

c + geom_area(stat = "bin")
x, y, alpha, color, fill, linetype, size

c + geom_density(kernel = "gaussian")
x, y, alpha, color, fill, group, linetype, size, weight

c + geom_dotplot()
x, y, alpha, color, fill

c + geom_freqpoly()
x, y, alpha, color, group, linetype, size

c + geom_histogram(binwidth = 5)
x, y, alpha, color, fill, linetype, size, weight

c2 + geom_qq(aes(sample = hwy))
x, y, alpha, color, fill, linetype, size, weight

Discrete

d <- ggplot(mpg, aes(fl))

d + geom_bar()
x, alpha, color, fill, linetype, size, weight

Two Variables

Continuous X, Continuous Y

e <- ggplot(mpg, aes(cty, hwy))

e + geom_label(aes(label = cty), nudge_x = 1, nudge_y = 1, check_overlap = TRUE)
x, y, label, alpha, angle, color, family, fontface, hjust, lineheight, size, vjust

e + geom_jitter(height = 2, width = 2)
x, y, alpha, color, fill, shape, size

e + geom_point()
x, y, alpha, color, fill, shape, size, stroke

e + geom_quantile()
x, y, alpha, color, group, linetype, size, weight

e + geom_rug(sides = "bl")
x, y, alpha, color, linetype, size

e + geom_smooth(method = lm)
x, y, alpha, color, fill, group, linetype, size, weight

e + geom_text(aes(label = cty), nudge_x = 1, nudge_y = 1, check_overlap = TRUE)
x, y, label, alpha, angle, color, family, fontface, hjust, lineheight, size, vjust

Discrete X, Continuous Y

f <- ggplot(mpg, aes(class, hwy))

f + geom_col()
x, y, alpha, color, fill, group, linetype, size

f + geom_boxplot()
x, y, lower, middle, upper, ymax, ymin, alpha, color, fill, group, linetype, shape, size, weight

f + geom_dotplot(binaxis = "y", stackdir = "center")
x, y, alpha, color, fill, group

f + geom_violin(scale = "area")
x, y, alpha, color, fill, group, linetype, size, weight

Discrete X, Discrete Y

g <- ggplot(diamonds, aes(cut, color))

g + geom_count()
x, y, alpha, color, fill, shape, size, stroke

Continuous Bivariate Distribution

h <- ggplot(diamonds, aes(carat, price))

h + geom_bin2d(binwidth = c(0.25, 500))
x, y, alpha, color, fill, linetype, size, weight

h + geom_density2d()
x, y, alpha, colour, group, linetype, size

h + geom_hex()
x, y, alpha, colour, fill, size

Continuous Function

i <- ggplot(economics, aes(date, unemploy))

i + geom_area()
x, y, alpha, color, fill, linetype, size

i + geom_line()
x, y, alpha, color, group, linetype, size

i + geom_step(direction = "hv")
x, y, alpha, color, group, linetype, size

Visualizing error

df <- data.frame(grp = c("A", "B"), fit = 4:5, se = 1:2)
j <- ggplot(df, aes(grp, fit, ymin = fit-se, ymax = fit+se))

j + geom_crossbar(fatten = 2)
x, y, ymax, ymin, alpha, color, fill, group, linetype, size

j + geom_errorbar()
x, ymax, ymin, alpha, color, group, linetype, size, width (also geom_errorbarh())

j + geom_linerange()
x, ymin, ymax, alpha, color, group, linetype, size

j + geom_pointrange()
x, y, ymin, ymax, alpha, color, fill, group, linetype, shape, size

Maps

data <- data.frame(murder = USArrests$Murder, state = tolower(rownames(USArrests)))
map <- map_data("state")
k <- ggplot(data, aes(fill = murder))

k + geom_map(aes(map_id = state), map = map) + expand_limits(x = map$long, y = map$lat)
map_id, alpha, color, fill, linetype, size

Three Variables

seals$z <- with(seals, sqrt(delta_long^2 + delta_lat^2))
l <- ggplot(seals, aes(long, lat))

l + geom_contour(aes(z = z))
x, y, z, alpha, colour, group, linetype, size, weight

l + geom_raster(aes(fill = z), hjust=0.5, vjust=0.5, interpolate=FALSE)
x, y, alpha, fill

l + geom_tile(aes(fill = z))
x, y, alpha, color, fill, linetype, size, width

RStudio® is a trademark of RStudio, Inc. • CC BY RStudio • info@rstudio.com • 844-448-1212 • rstudio.com

Learn more at docs.ggplot2.org and www.ggplot2-exts.org • ggplot2 2.1.0 • Updated 11/16

https://www.rstudio.com/resources/cheatsheets/

# Repos of figures + code

- R graph catalog
- From Data to Viz
- The R graph gallery
- R graph gallery
- Cookbook for R: Graphs
- Graphical data analysis with R
- IEG figures

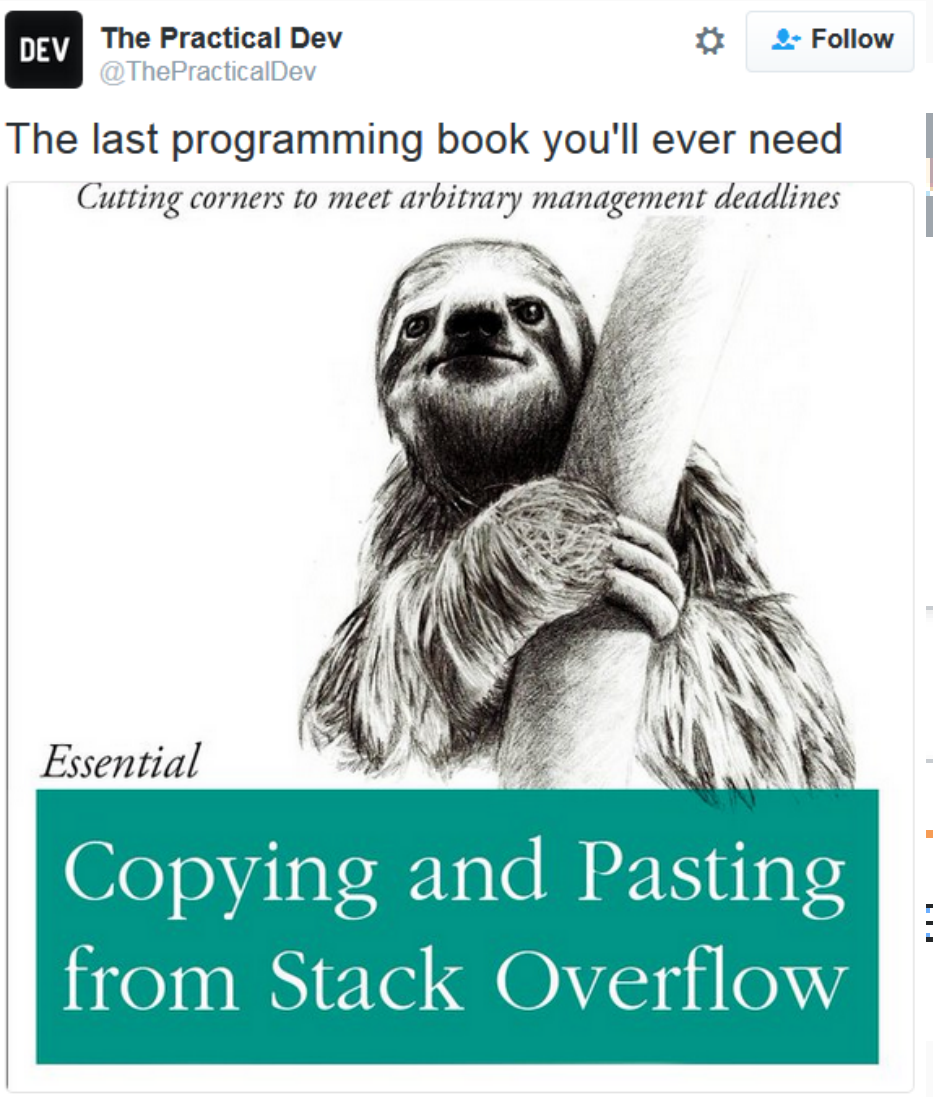
# Find answers for all your questions in Stack Overflow

stackoverflow

Search

ggplot2

36,854 results



# Building a ggplot figure

# Our example dataset: paper planes flying experiment

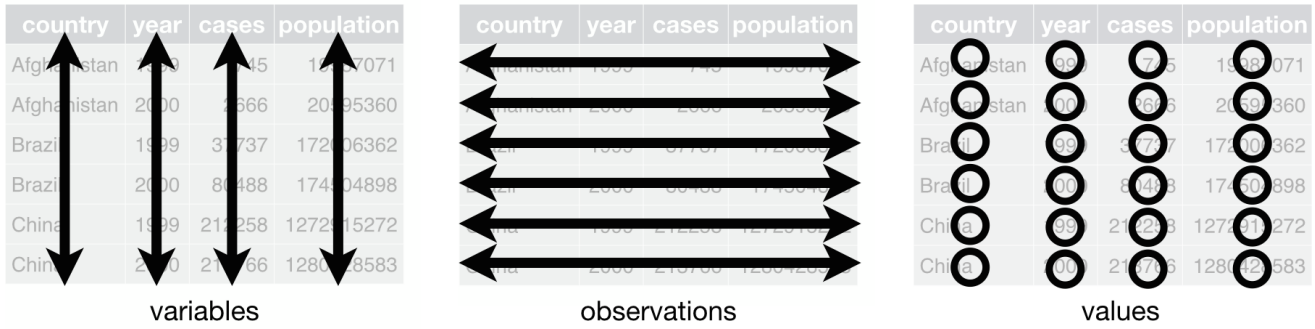
```
library(paperplanes)
head(paperplanes)
```

| id | hour | person | gender | age | plane | paper | distance |
|---|---|---|---|---|---|---|---|
| 1 | [17,18) | Roland | male | 30 | Standard80 | 80 | 7.8 |
| 2 | [17,18) | Astrid | female | 30 | Concorde120 | 120 | 2.7 |
| 3 | [17,18) | Roland | male | 30 | Standard120 | 120 | 9.2 |
| 4 | [17,18) | Isabella | female | 48 | Standard120 | 120 | 6.0 |
| 5 | [17,18) | Fabienne | female | 17 | Standard120 | 120 | 7.3 |
| 6 | [17,18) | Fabienne | female | 17 | Standard120 | 120 | 7.8 |

Ensuring paper is factor, not numeric

```
paperplanes$paper <- as.factor(paperplanes$paper)
```

# Data must be a tidy data frame




**tidyr::gather(table4, key = "year", value = "cases", "1999", "2000")**

| country | year | cases |
|---|---|---|
| Afghanistan | 1999 | 745 |
| Afghanistan | 2000 | 2666 |
| Brazil | 1999 | 37737 |
| Brazil | 2000 | 80488 |
| China | 1999 | 212258 |
| China | 2000 | 213766 |

| country | 1999 | 2000 |
|---|---|---|
| Afghanistan | 745 | 2666 |
| Brazil | 37737 | 80488 |
| China | 212258 | 213766 |

table4

http://r4ds.had.co.nz/tidy-data.html

# Calling ggplot

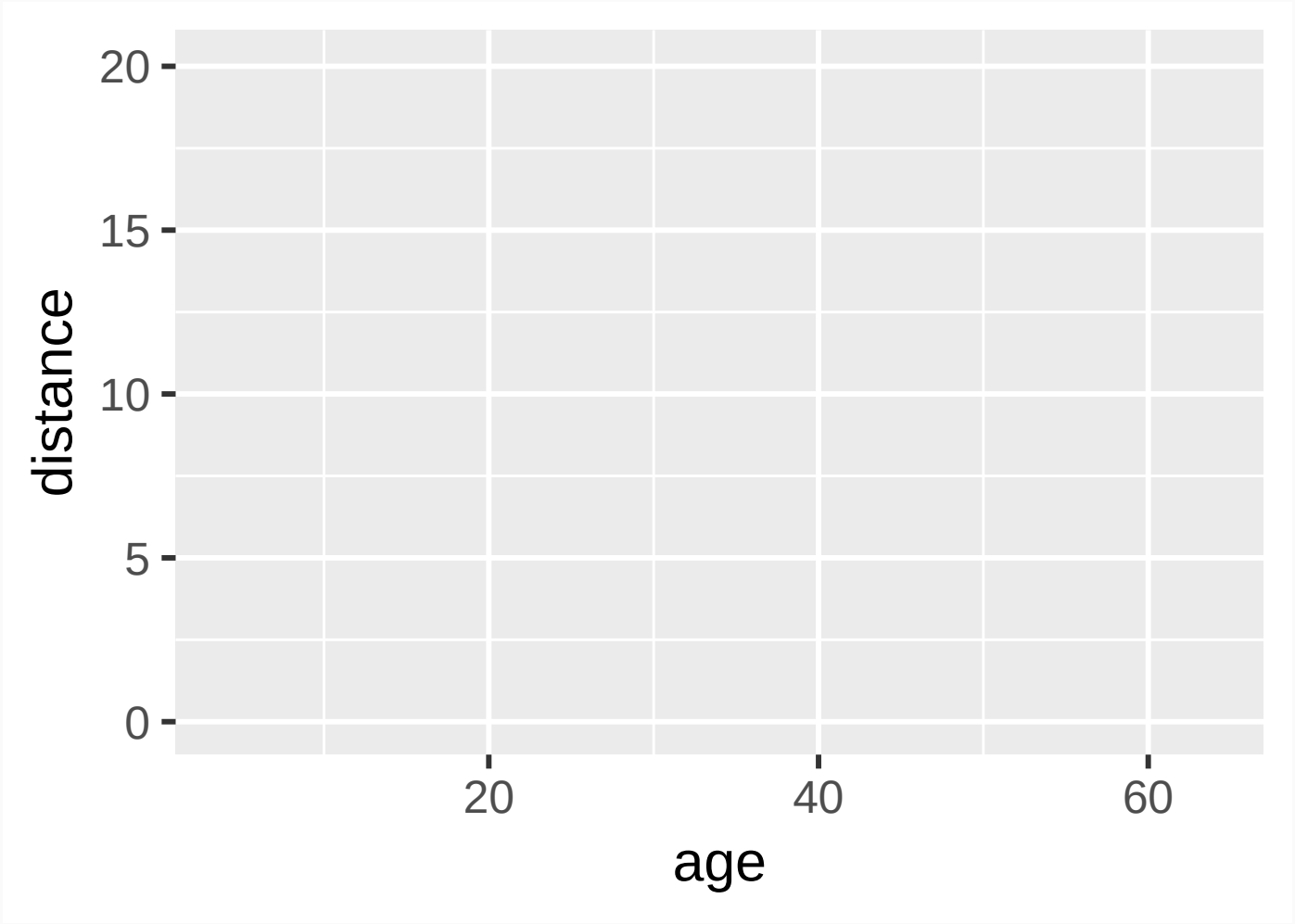
```
library(ggplot2)
ggplot(paperplanes)
```

```
ggplot(paperplanes)
```

First argument is a tidy data frame

# What variables as axes?

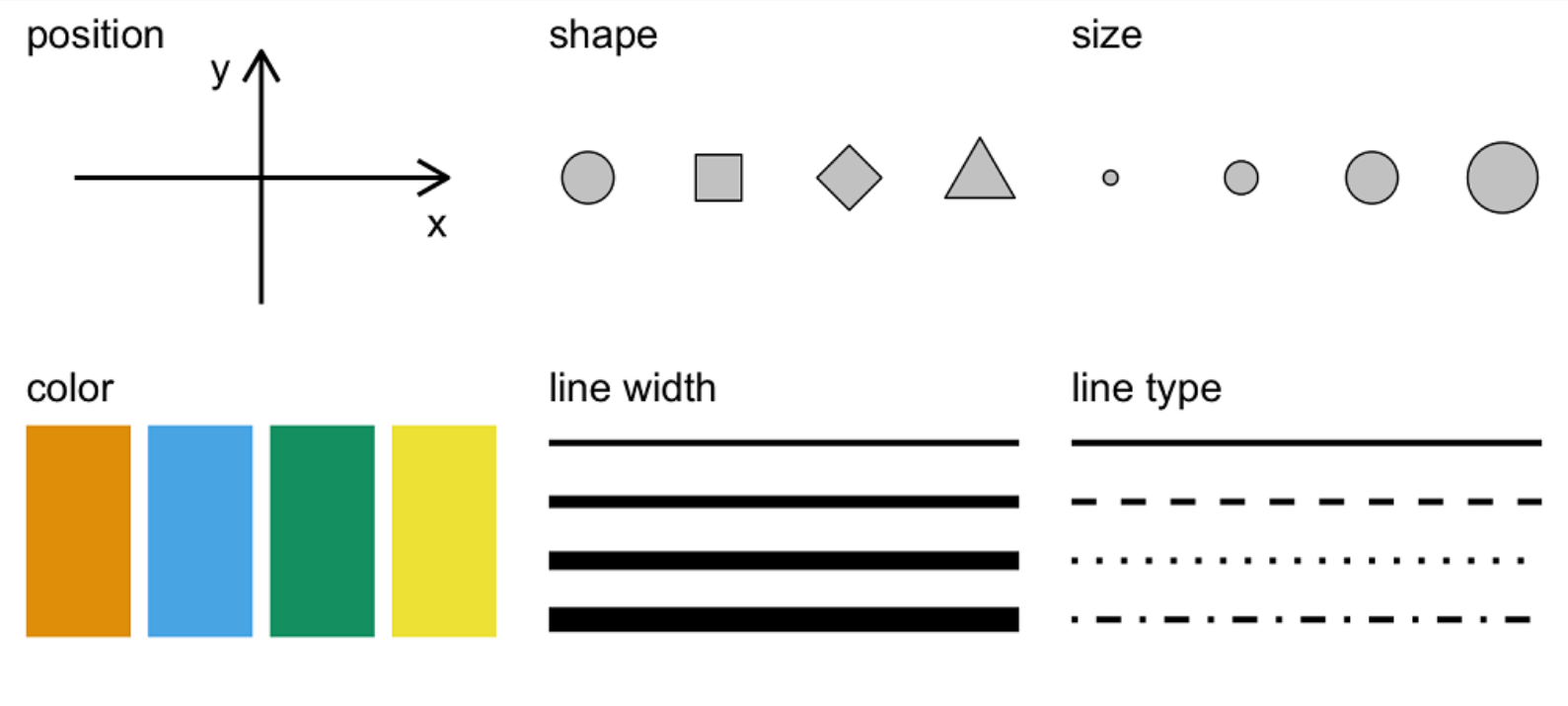
```
ggplot(paperplanes) +
  aes(x = age, y = distance)
```




Note syntax: + followed by new line

```
ggplot(paperplanes) +
  aes(x = age, y = distance)
```

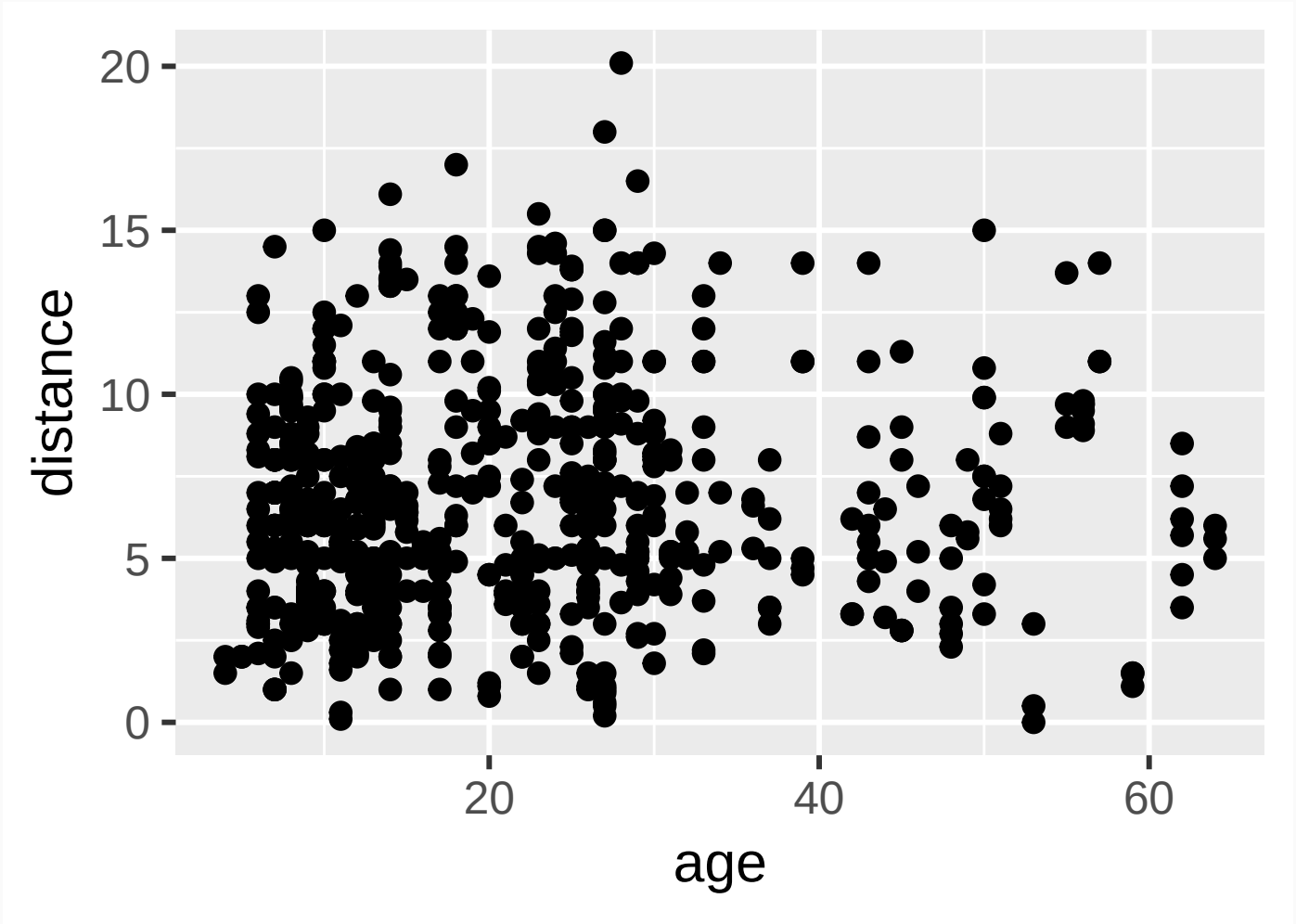
**Aesthetics** (aes) map data variables (*age*, *distance*) to graphic elements (*axes*)




http://serialmentor.com/dataviz/aesthetic-mapping.html

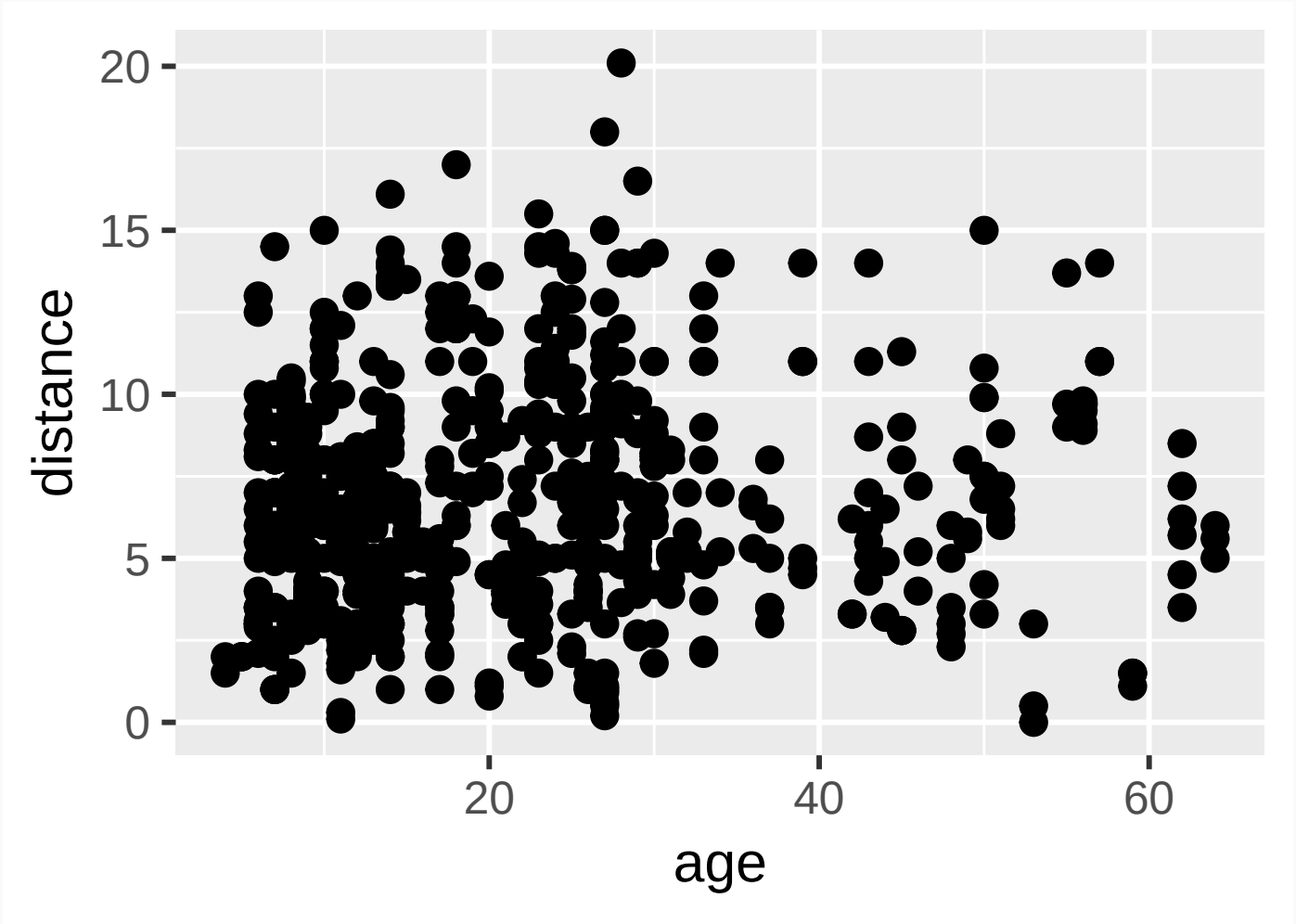
# Adding layers (geoms)

```
ggplot(paperplanes) +
  aes(x = age, y = distance) +
  geom_point()
```



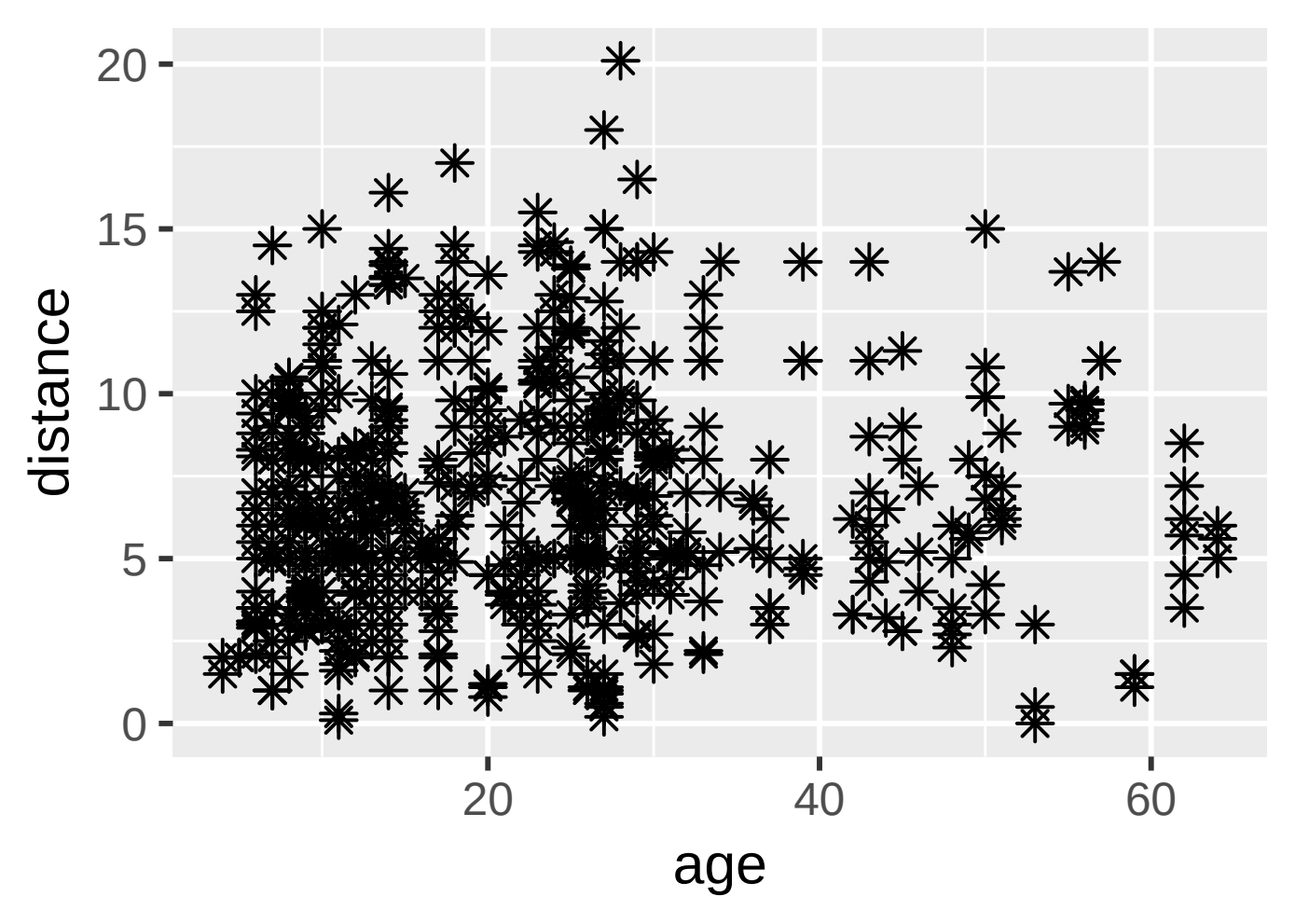
# Changing point size and type

```
ggplot(paperplanes) +
  aes(x = age, y = distance) +
  geom_point(size = 2)
```



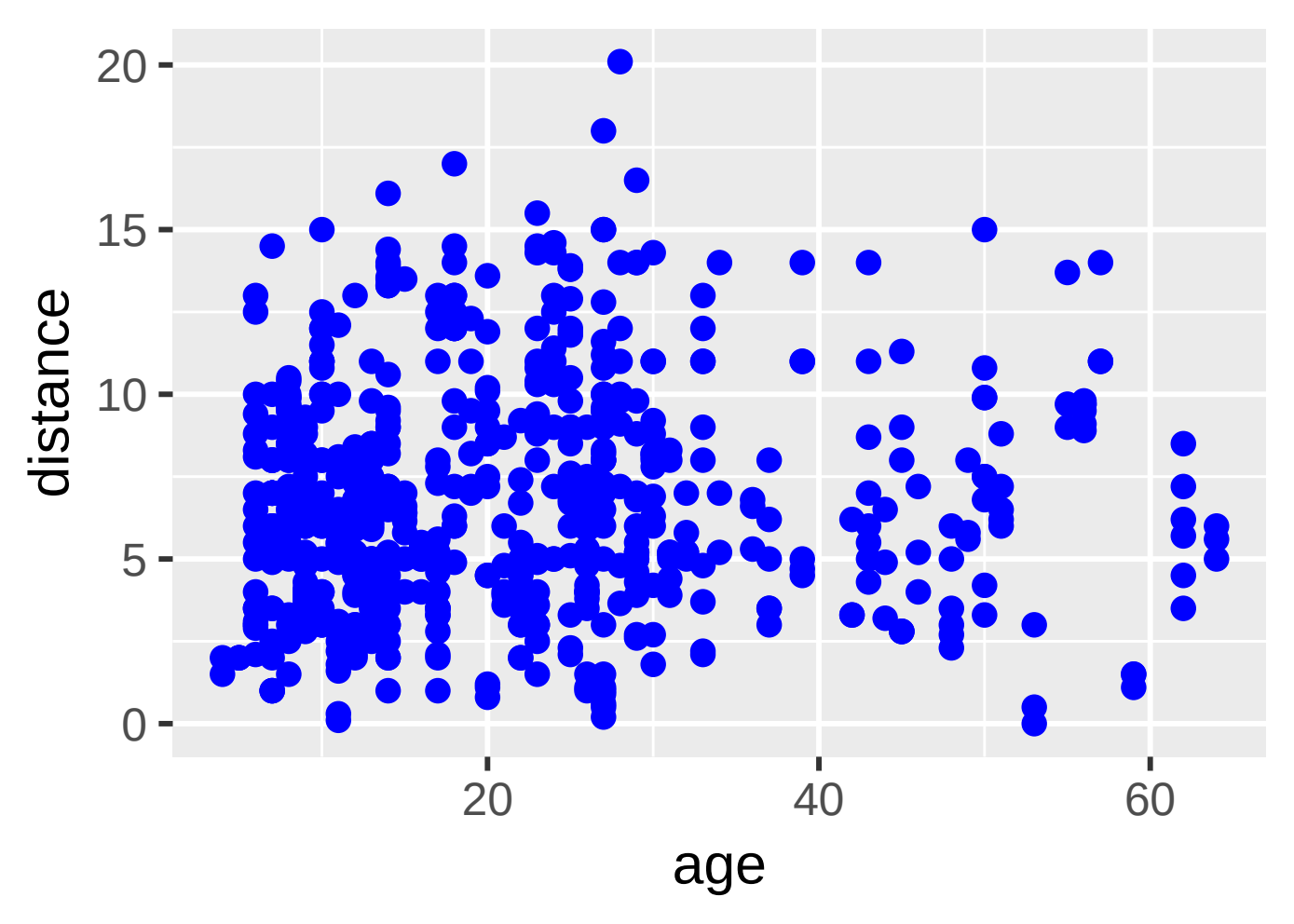
# Changing point size and type

```
ggplot(paperplanes) +
  aes(x = age, y = distance) +
  geom_point(size = 2, shape = 8)
```



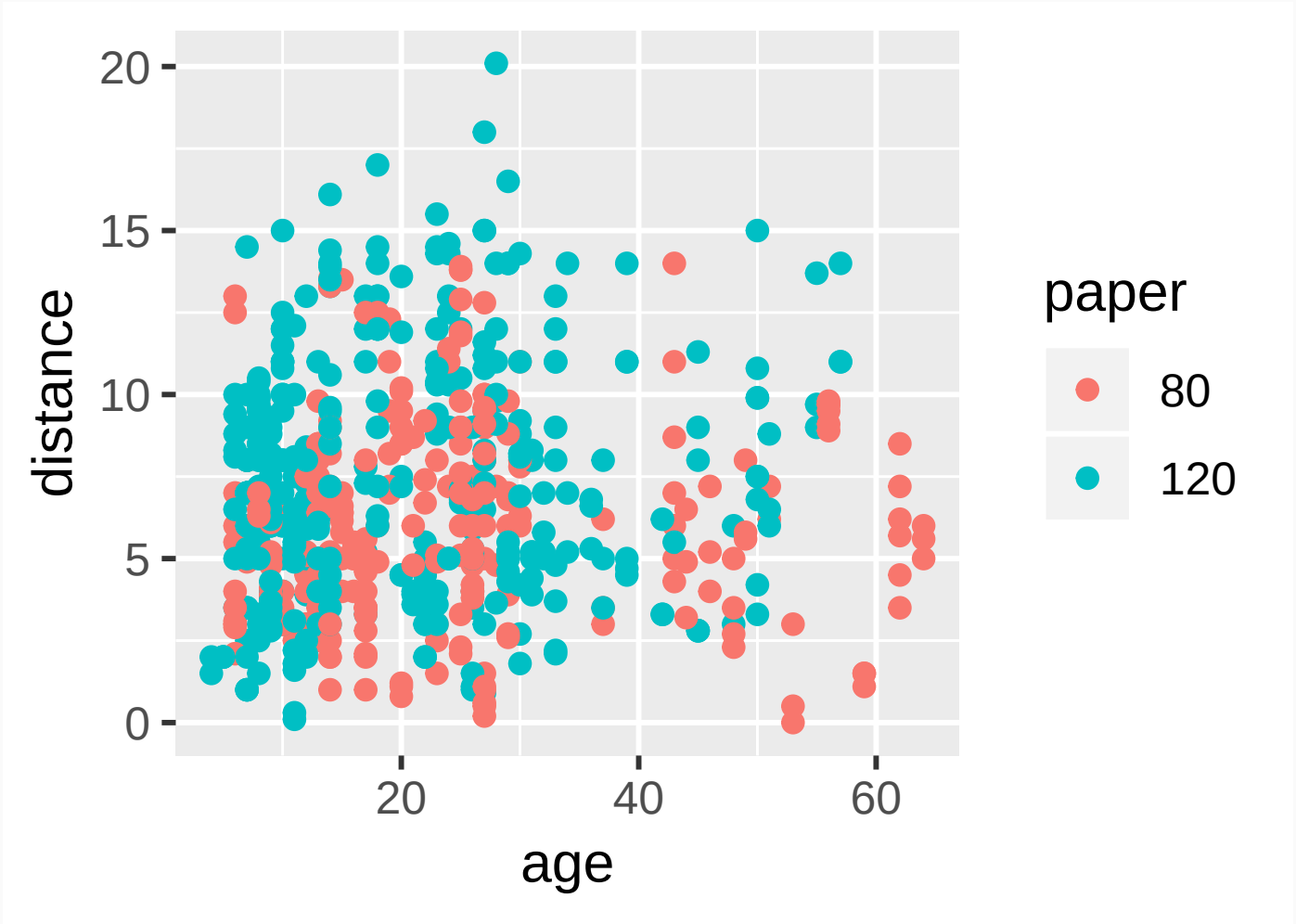
# Changing point size and type

```
ggplot(paperplanes) +
  aes(x = age, y = distance) +
  geom_point(size = 2, shape = 16, colour = "blue")
```



# Map geom aesthetics (e.g. colour) to variable

```
ggplot(paperplanes) +
  aes(x = age, y = distance) +
  geom_point(aes(colour = paper))
```



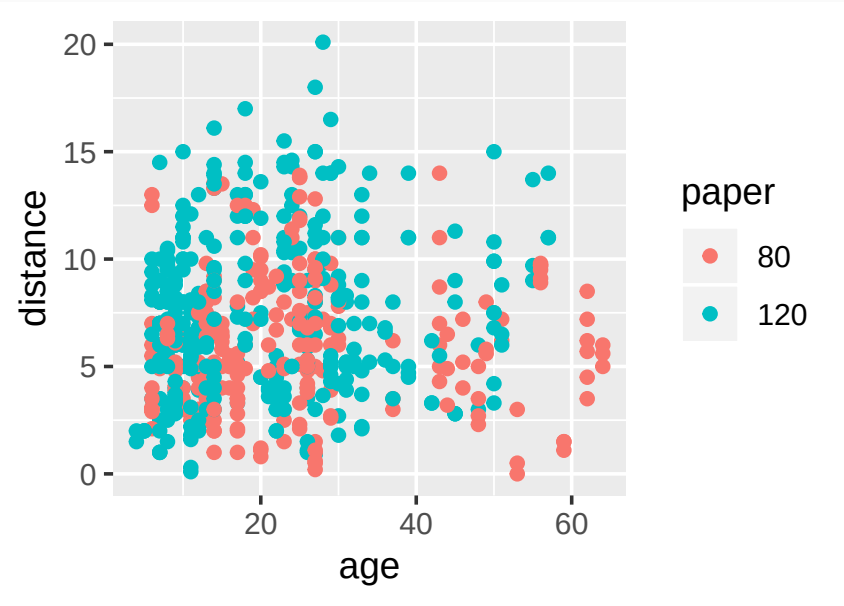
Note difference between

```
geom_point(colour = "blue")
# colour is given a concrete value ('blue')
```

```
geom_point(aes(colour = gender))
# colour maps a *variable* (using `aes`)
```

This works:

```
ggplot(paperplanes) +
  aes(x = age, y = distance) +
  geom_point(aes(colour = paper))
```




This doesn't work:

```
ggplot(paperplanes) +
  aes(x = age, y = distance) +
  geom_point(colour = paper)
```

*Error in layer(data = data, mapping = mapping, stat = stat, geom = GeomPoint, : object 'paper' not found*

.

.

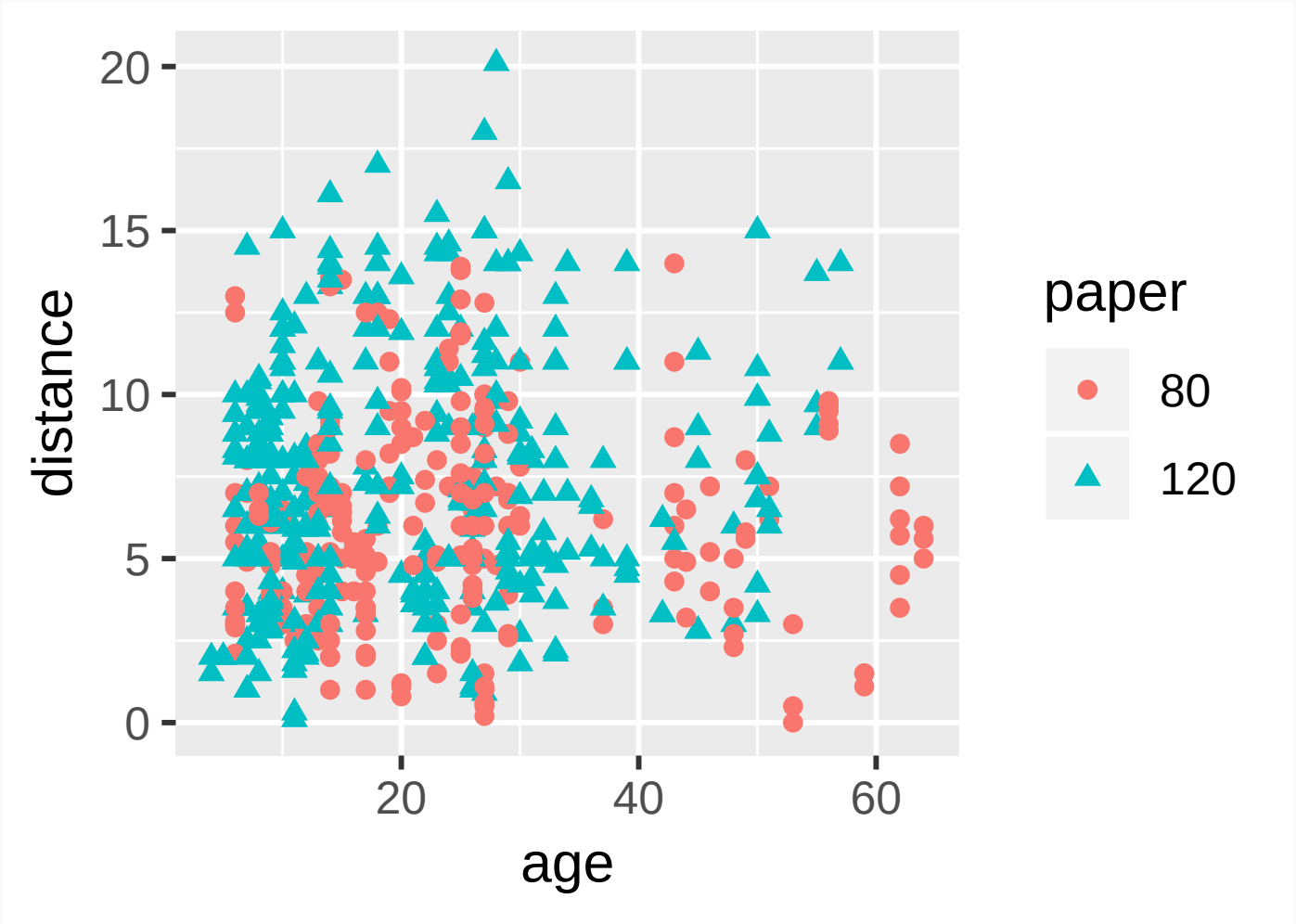
'paper' is a variable in dataframe

.

**Must use aes**

# Map geom aesthetics (colour, shape) to variable

```
ggplot(paperplanes) +
  aes(x = age, y = distance) +
  geom_point(aes(colour = paper, shape = paper))
```

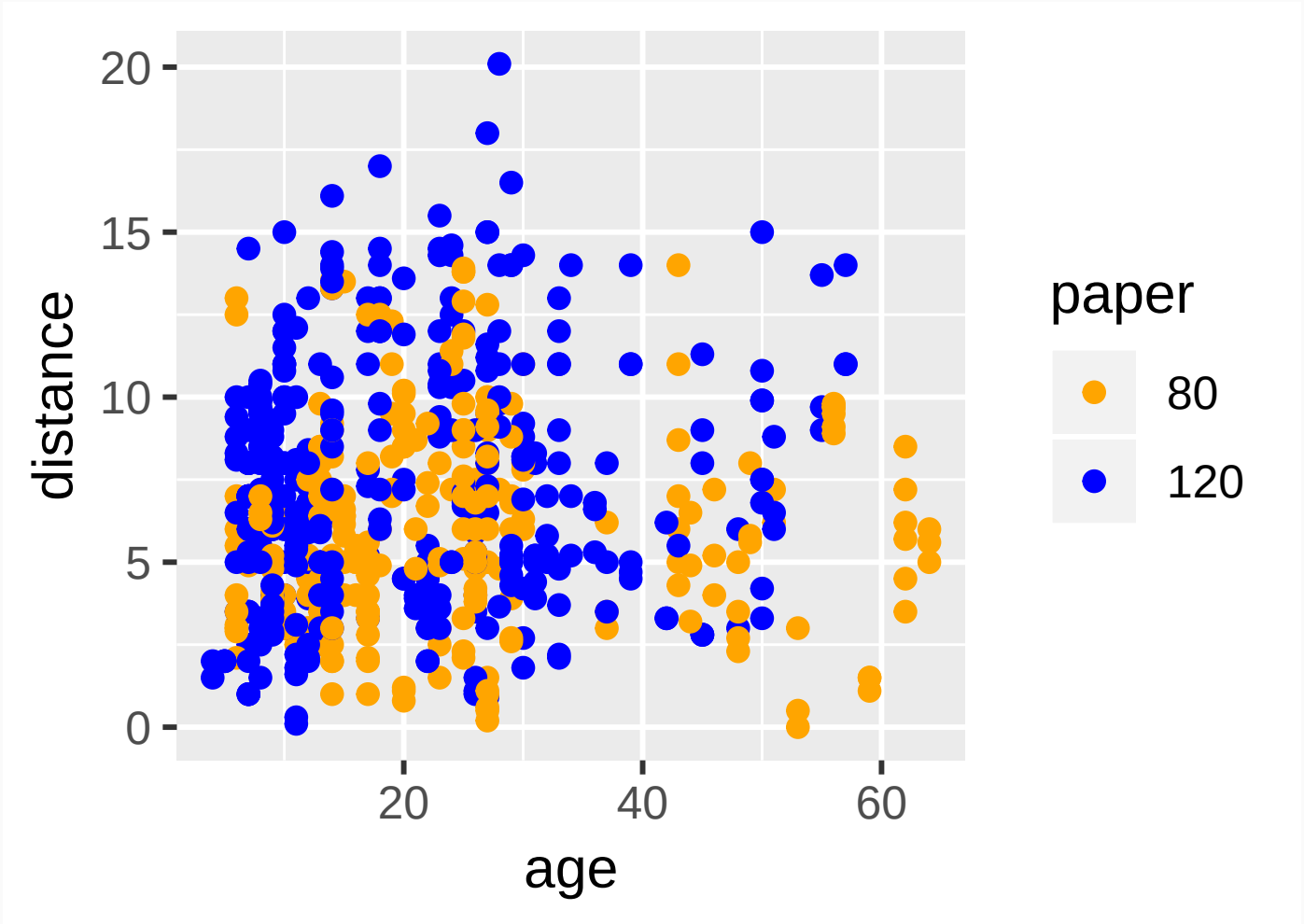


# Map geom aesthetics (colour, shape) to variable

```
ggplot(paperplanes) +
  aes(x = age, y = distance) +
  geom_point(aes(colour = paper, shape = gender))
```

# Change colour scale

```
ggplot(paperplanes) +
  aes(x = age, y = distance) +
  geom_point(aes(colour = paper)) +
  scale_colour_manual(values = c("orange", "blue"))
```

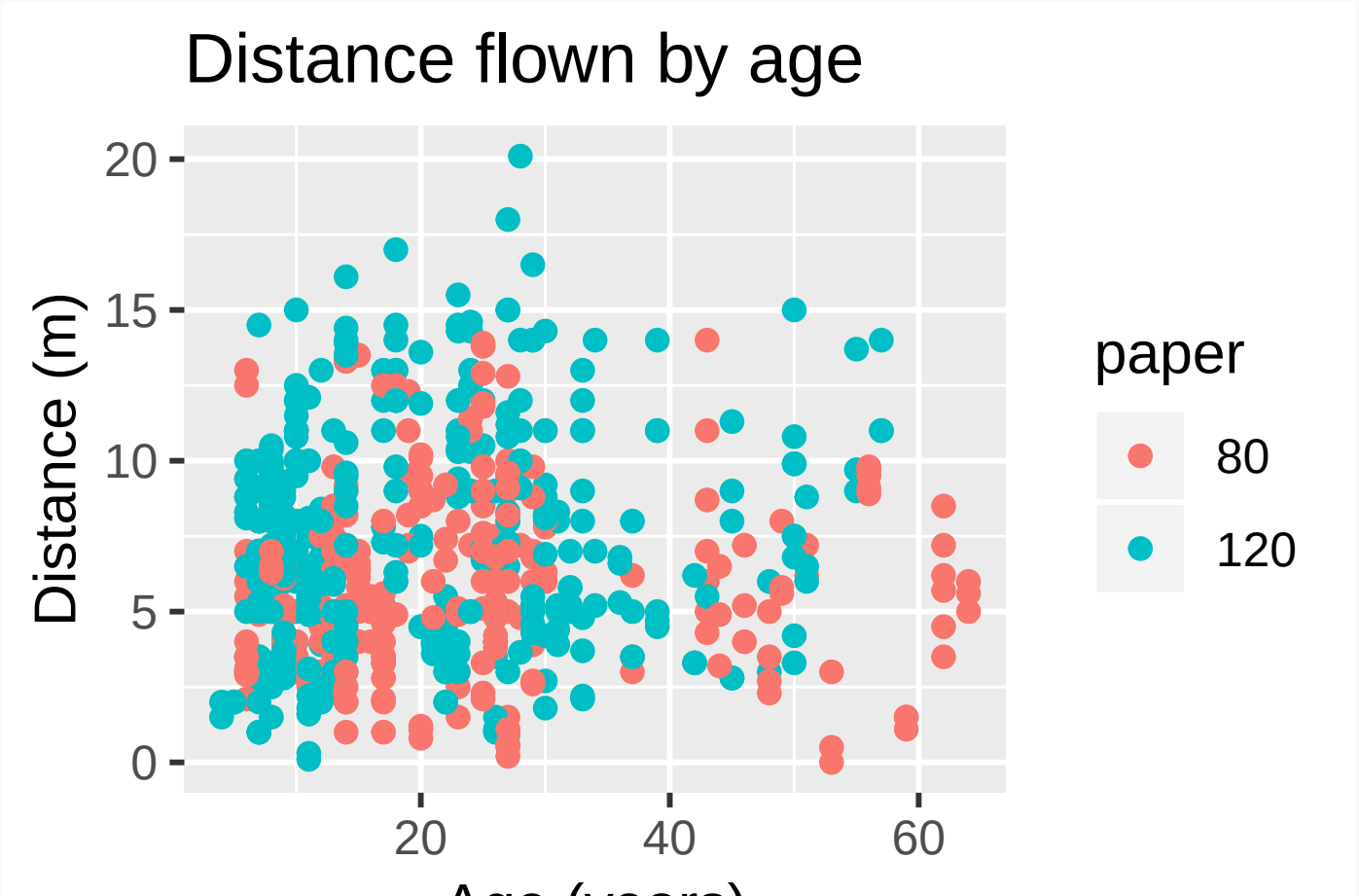


# Change axis labels: xlab & ylab

```
ggplot(paperplanes) +
  aes(x = age, y = distance) +
  geom_point(aes(colour = paper)) +
  labs(x = "Age (years)", y = "Distance (m)")
```

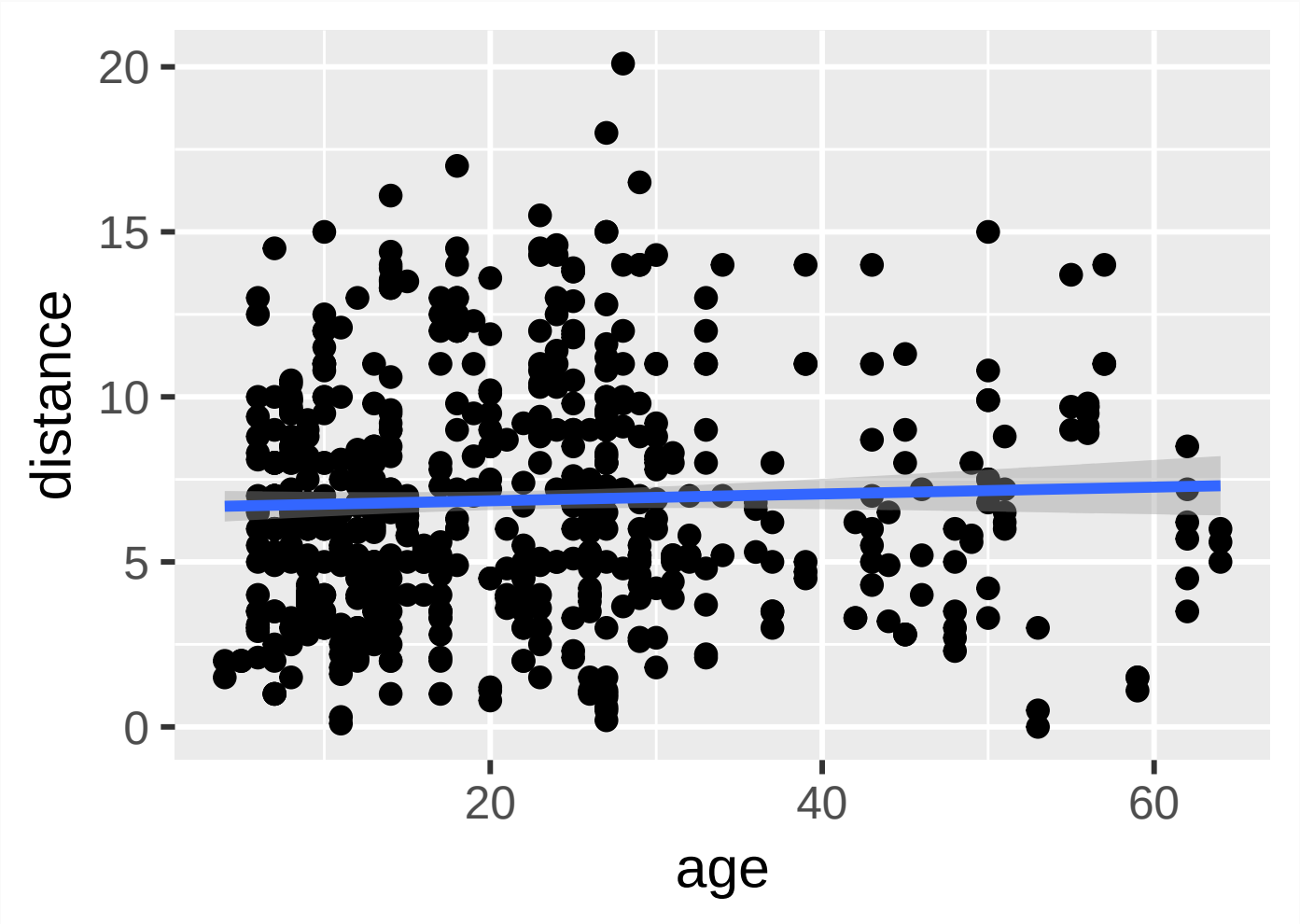
# Set title

```
ggplot(paperplanes) +
  aes(x = age, y = distance) +
  geom_point(aes(colour = paper)) +
  labs(x = "Age (years)", y = "Distance (m)") +
  labs(title = "Distance flown by age")
```



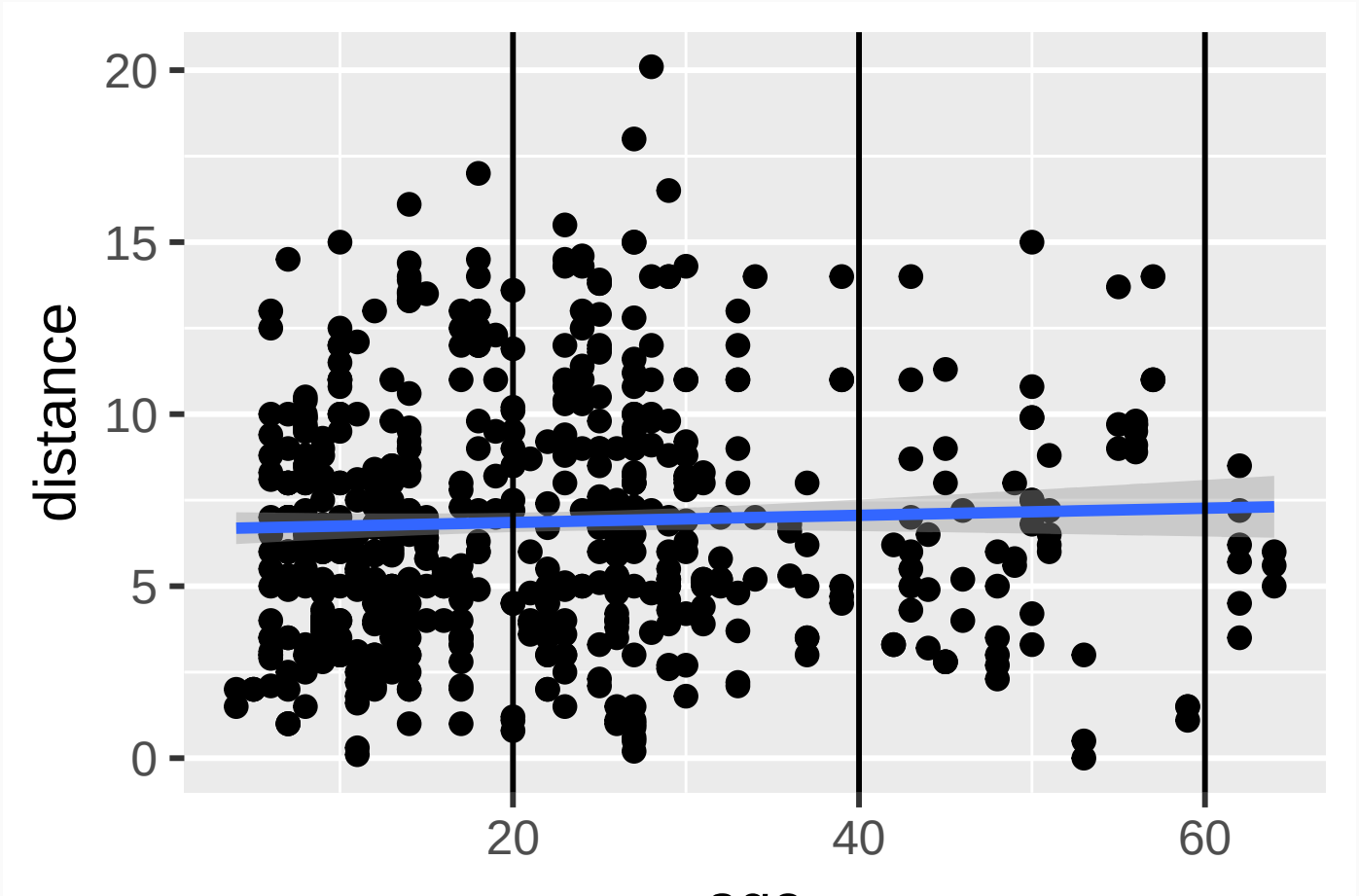
# Adding more layers

```
ggplot(paperplanes) +
  aes(x = age, y = distance) +
  geom_point() +
  geom_smooth(method = "lm")
```



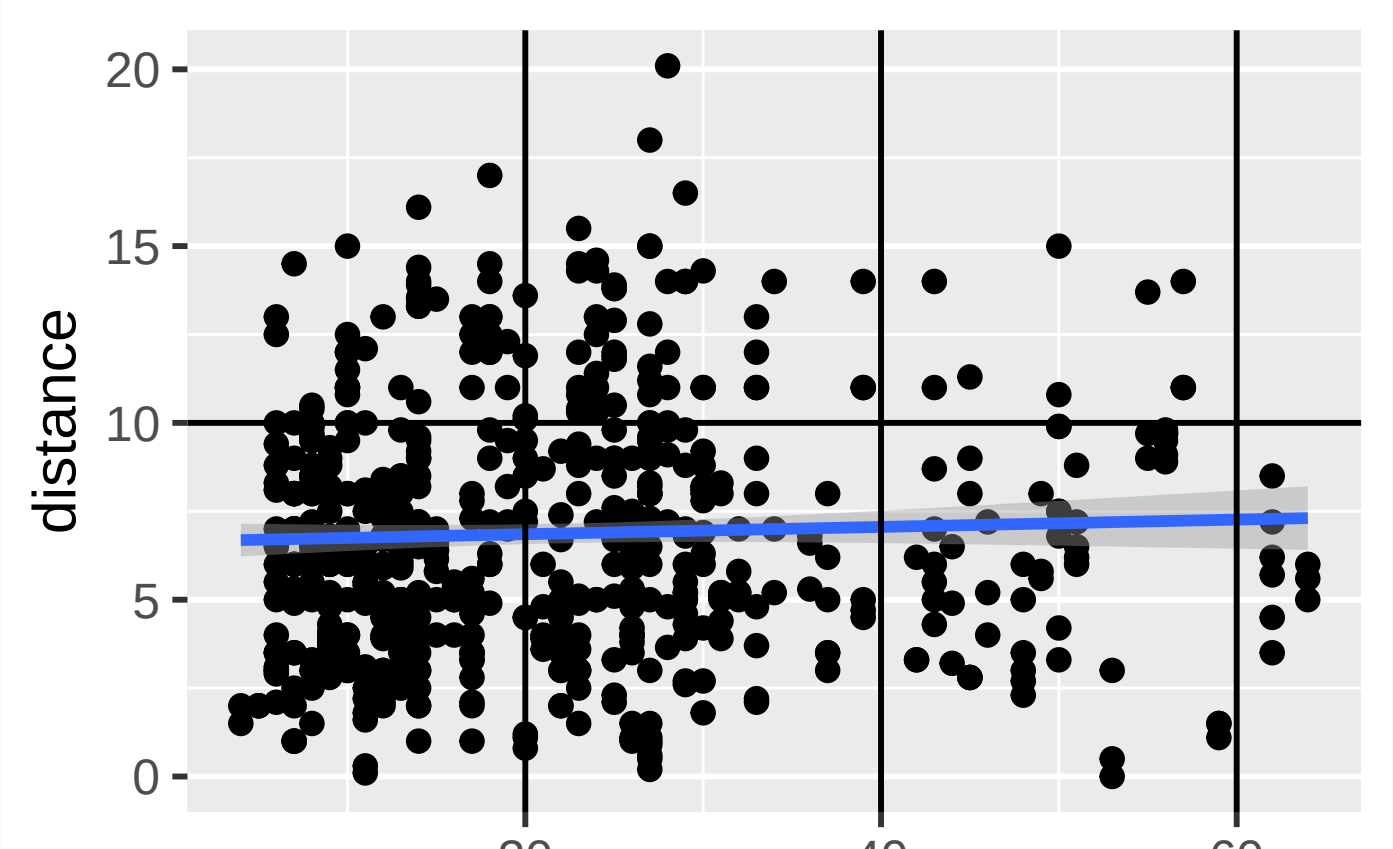
# Adding more layers

```
ggplot(paperplanes) +
  aes(x = age, y = distance) +
  geom_point() +
  geom_smooth(method = "lm") +
  geom_vline(xintercept = c(20, 40, 60))
```



# Adding more layers

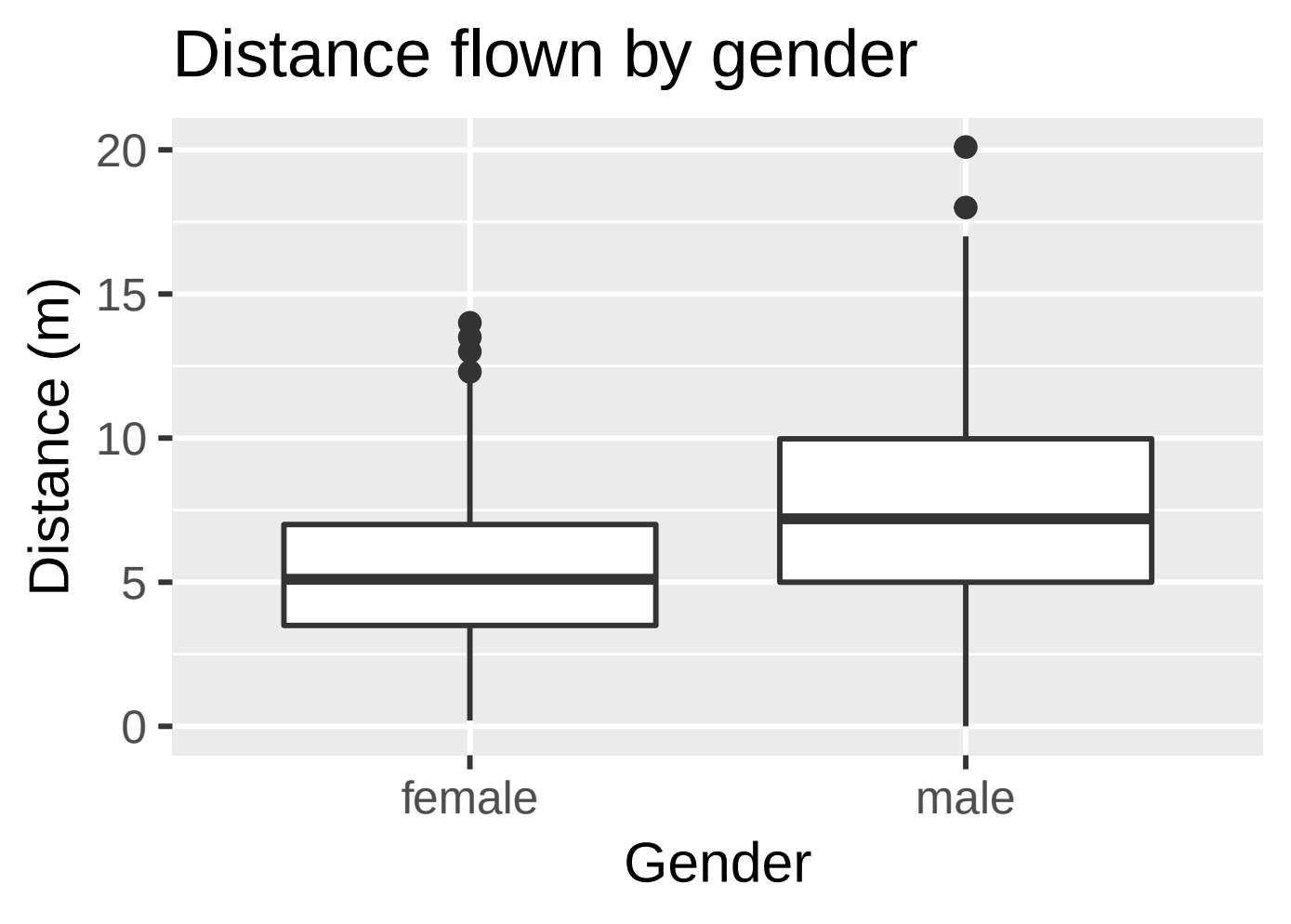
```
ggplot(paperplanes) +
  aes(x = age, y = distance) +
  geom_point() +
  geom_smooth(method = "lm") +
  geom_vline(xintercept = c(20, 40, 60)) +
  geom_hline(yintercept = 10)
```



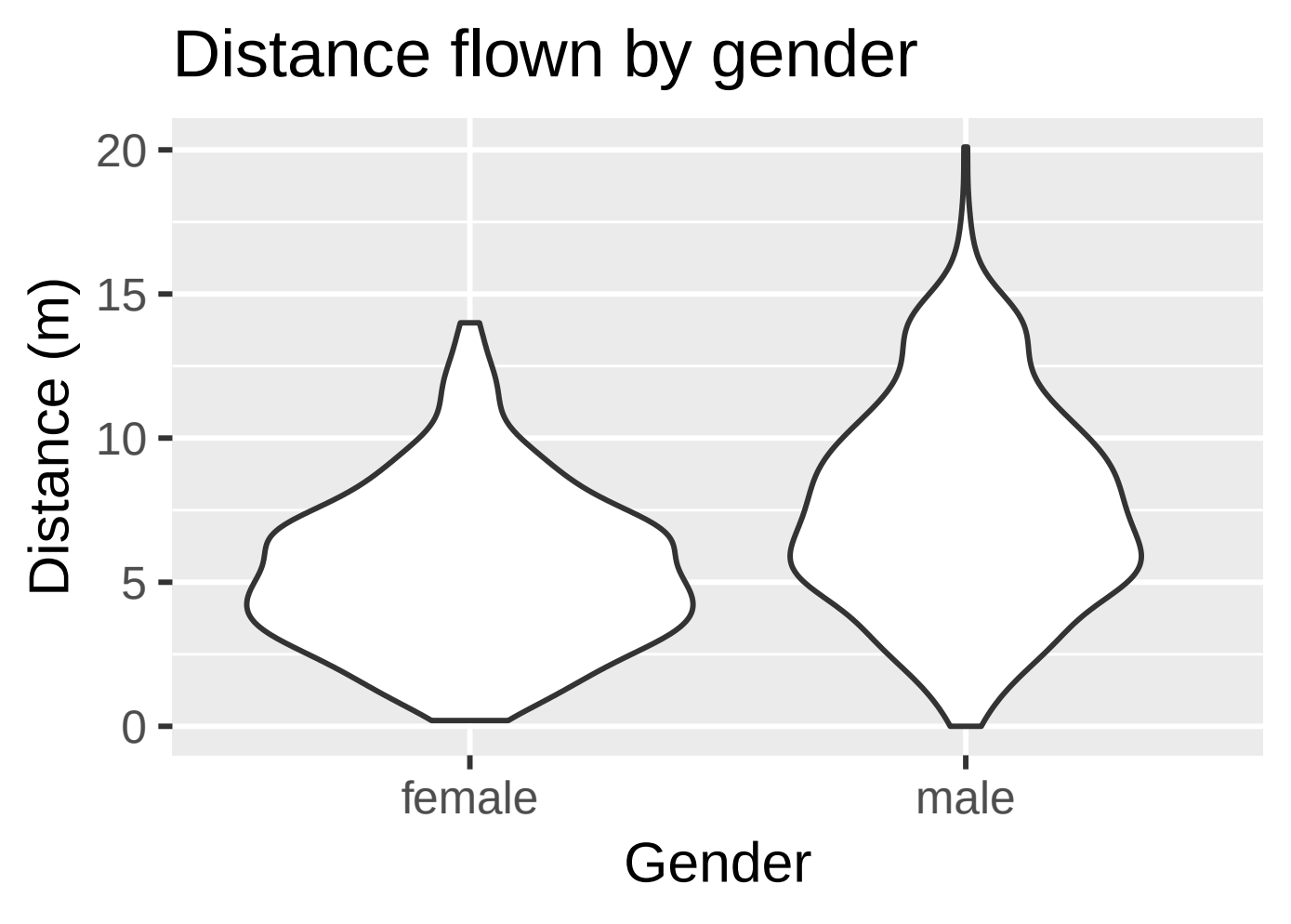
# Summary

```
ggplot(paperplanes) +           # Name of (tidy) data frame
  aes(x = age, y = distance) + # Aesthetics (variables to map in axes)
  geom_point()                  # Geoms: geometric objects
```

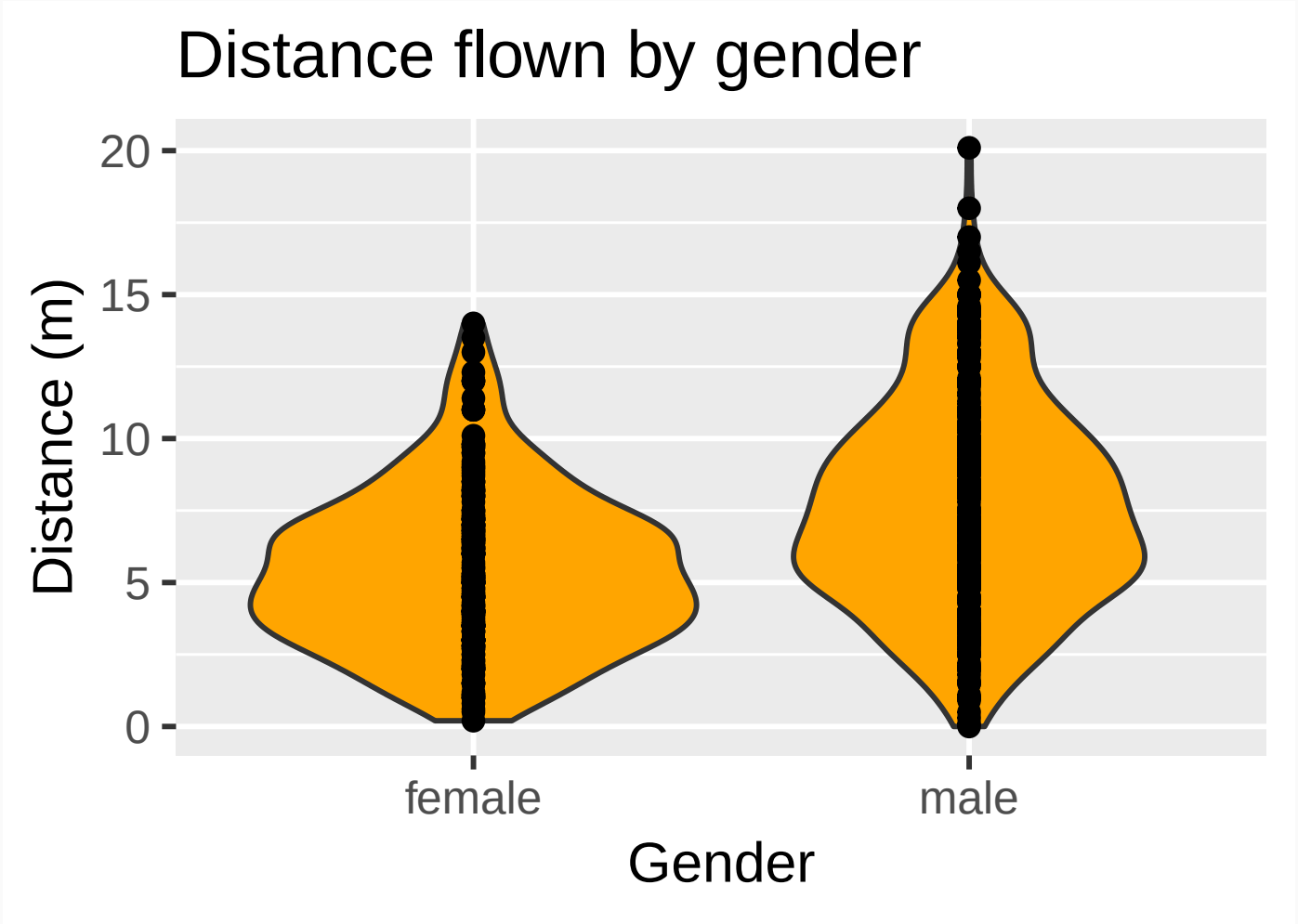
# Exercise: Make a plot like this one



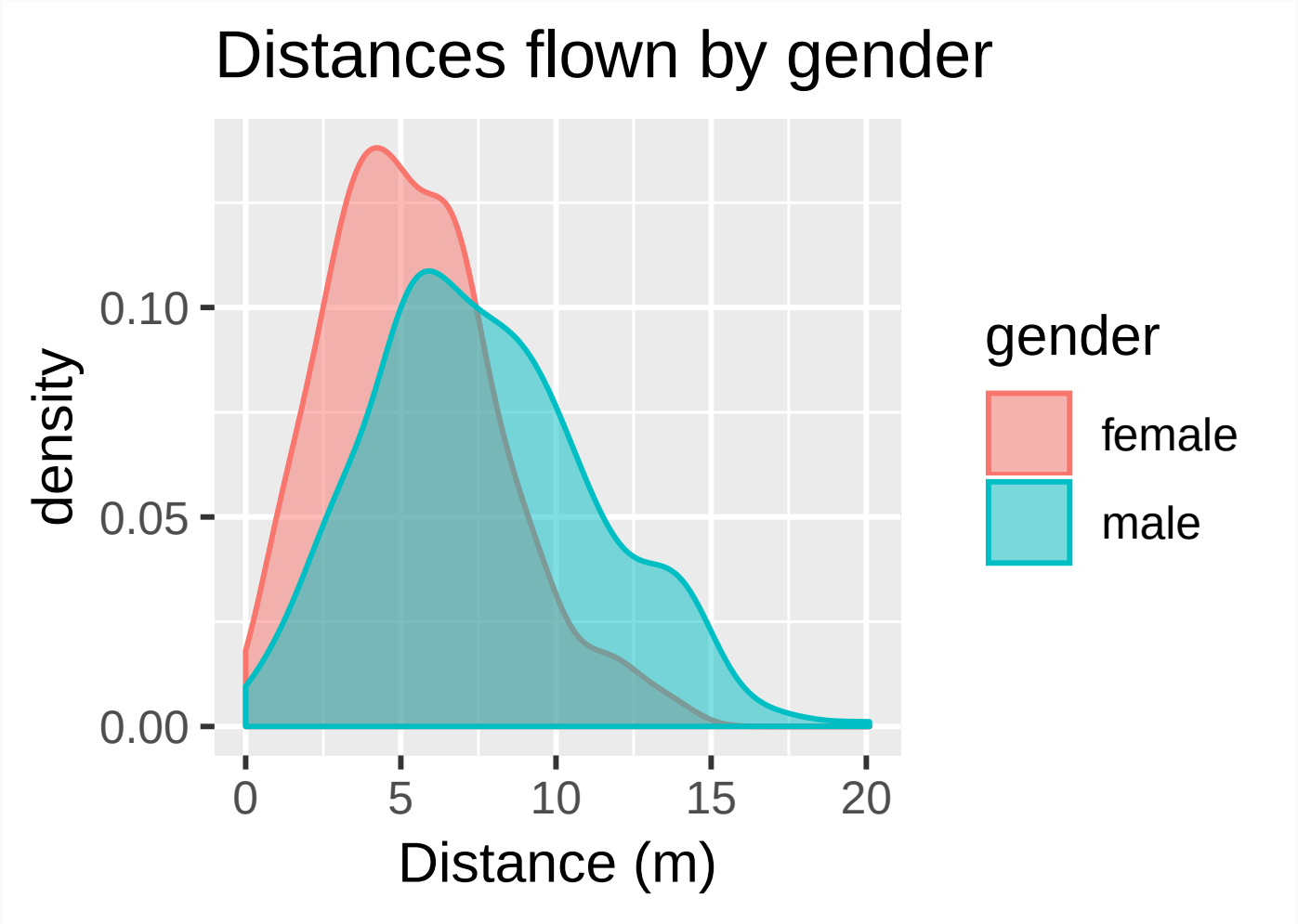
# Exercise: Make a plot like this one



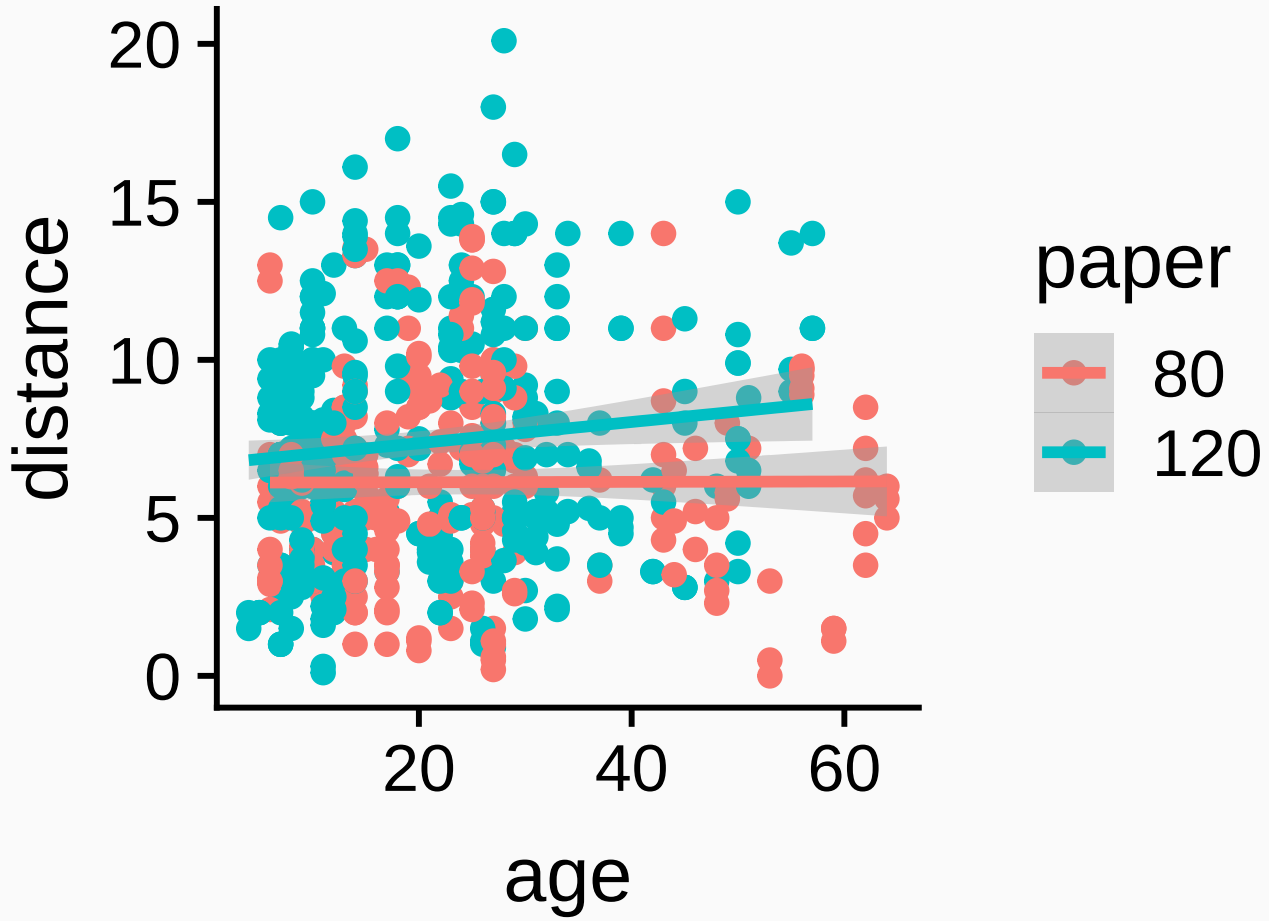
# Exercise: Make a plot like this one



# Exercise: Make a plot like this one



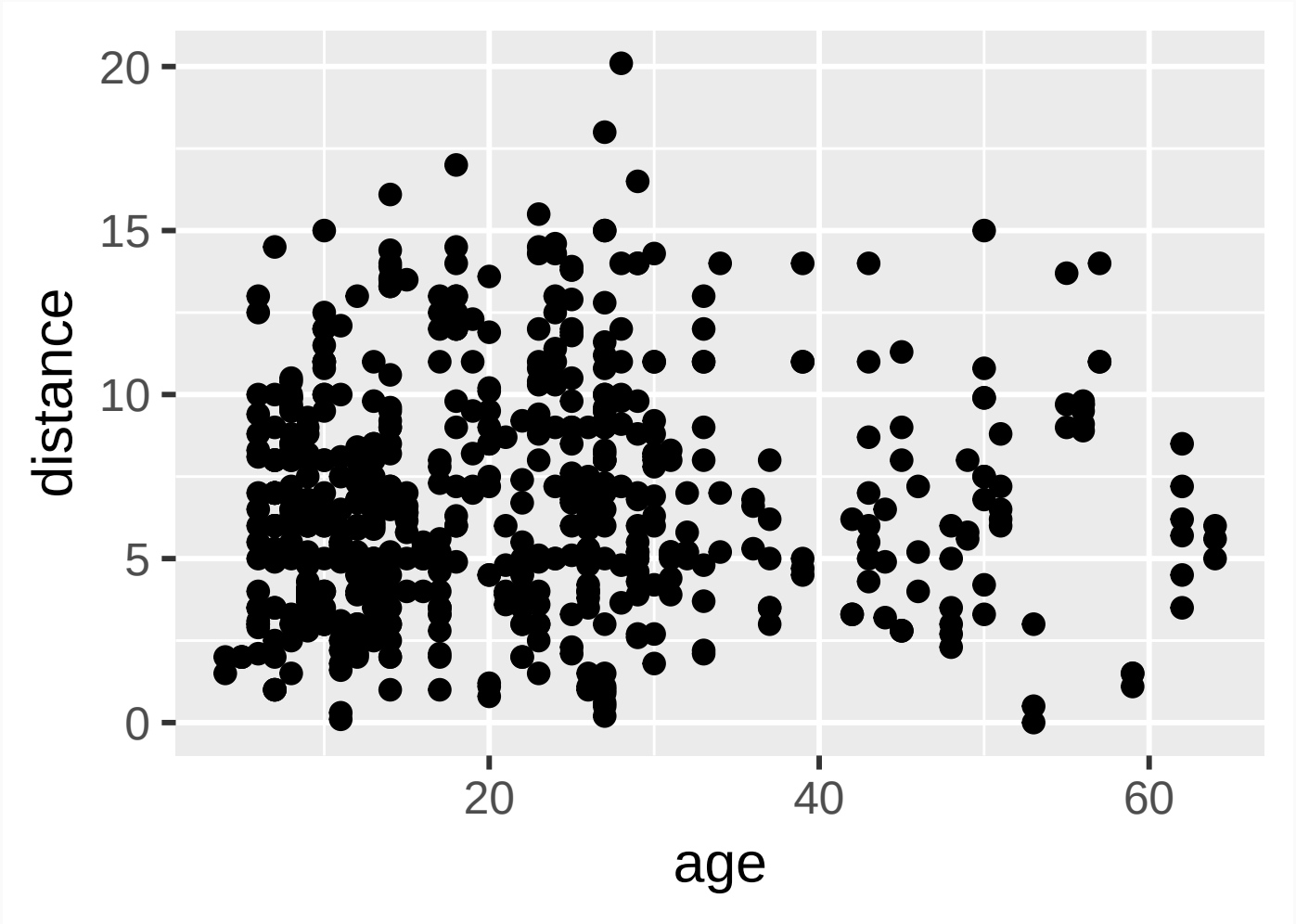
# Exercise: Make a plot like this one



## ggplot2 figures can be assigned to R objects

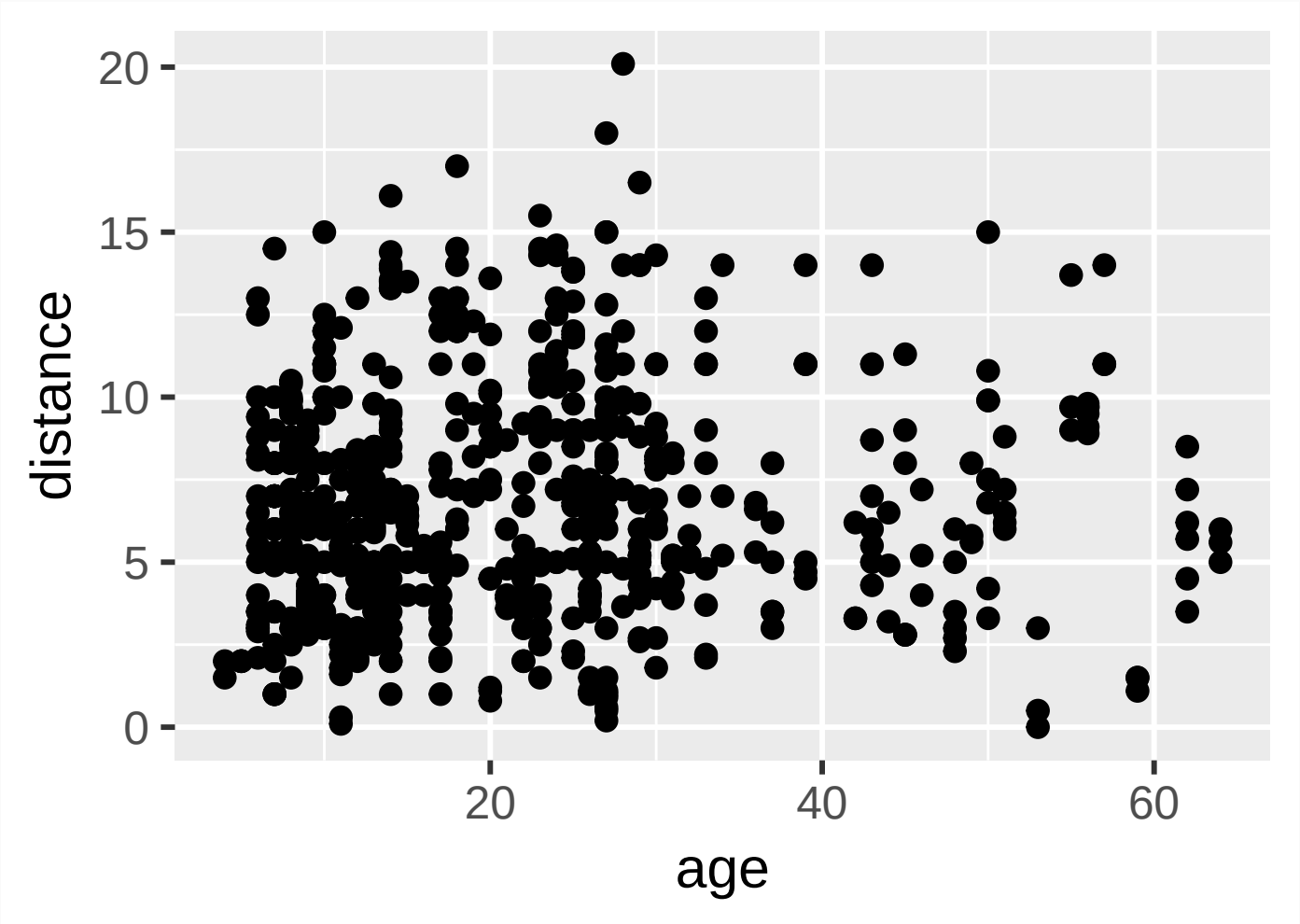
# Assigning ggplot objects

```
myplot <- ggplot(paperplanes) +
  aes(x = age, y = distance)
myplot + geom_point()
```



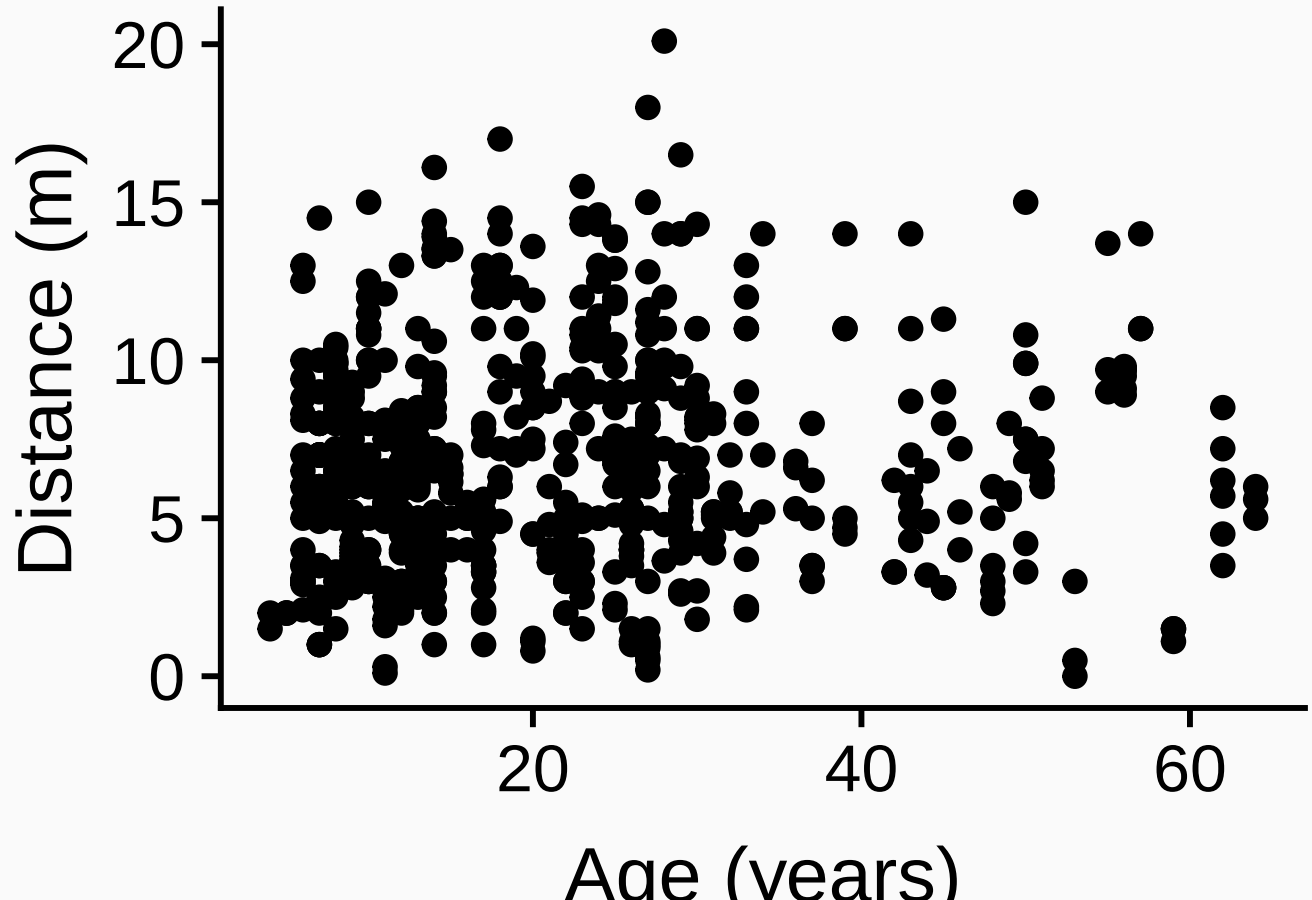
# Assigning ggplot objects

```
myplot <- ggplot(paperplanes) +
  aes(x = age, y = distance)
myplot <- myplot + geom_point()
myplot
```



# Assigning ggplot objects

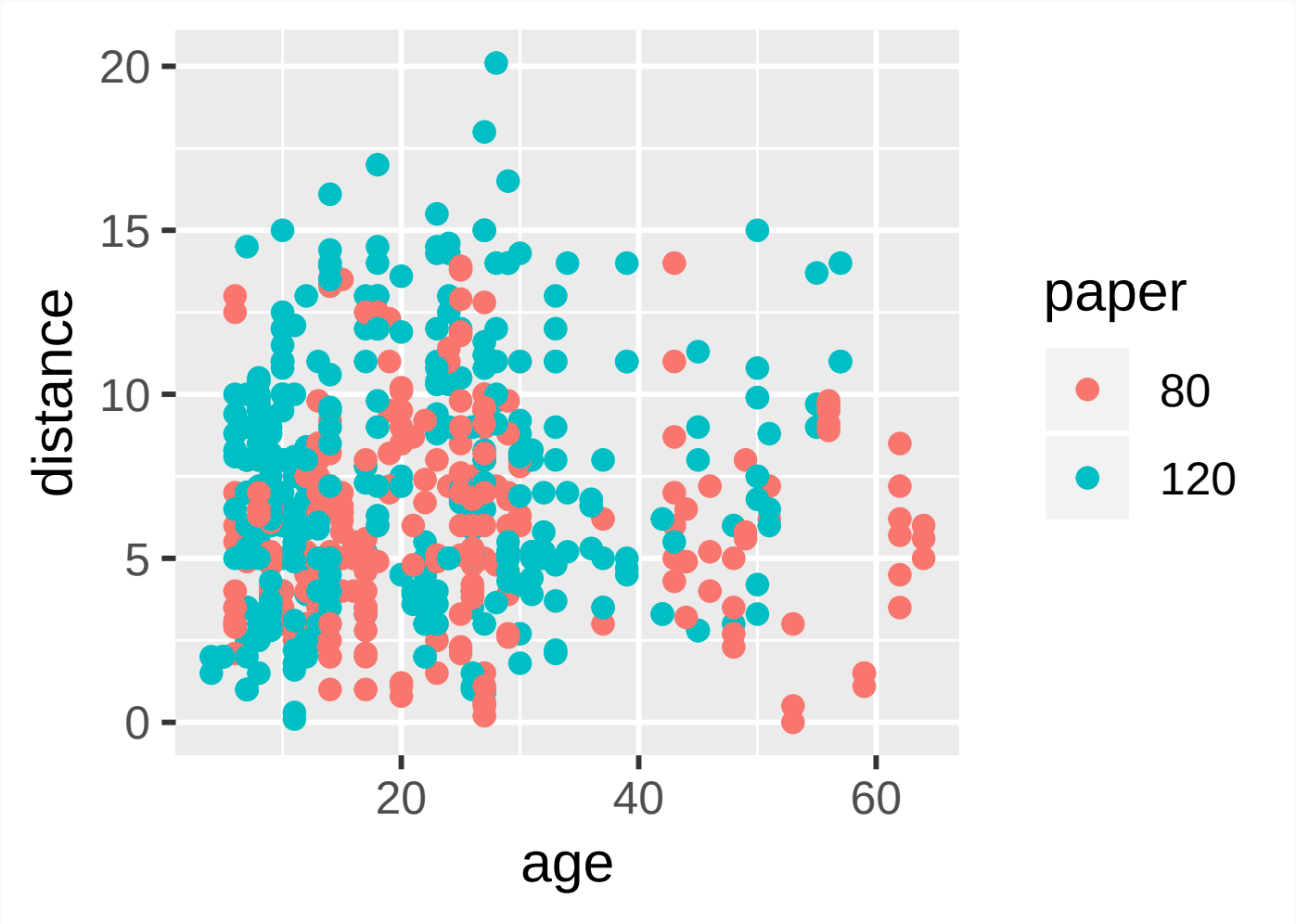
```
baseplot <- ggplot(paperplanes) +
  aes(x = age, y = distance)
scatterplot <- baseplot + geom_point()
labelled <- scatterplot + labs(x = "Age (years)", y = "Distance (m)")
labelled
```



# Themes: changing plot appearance

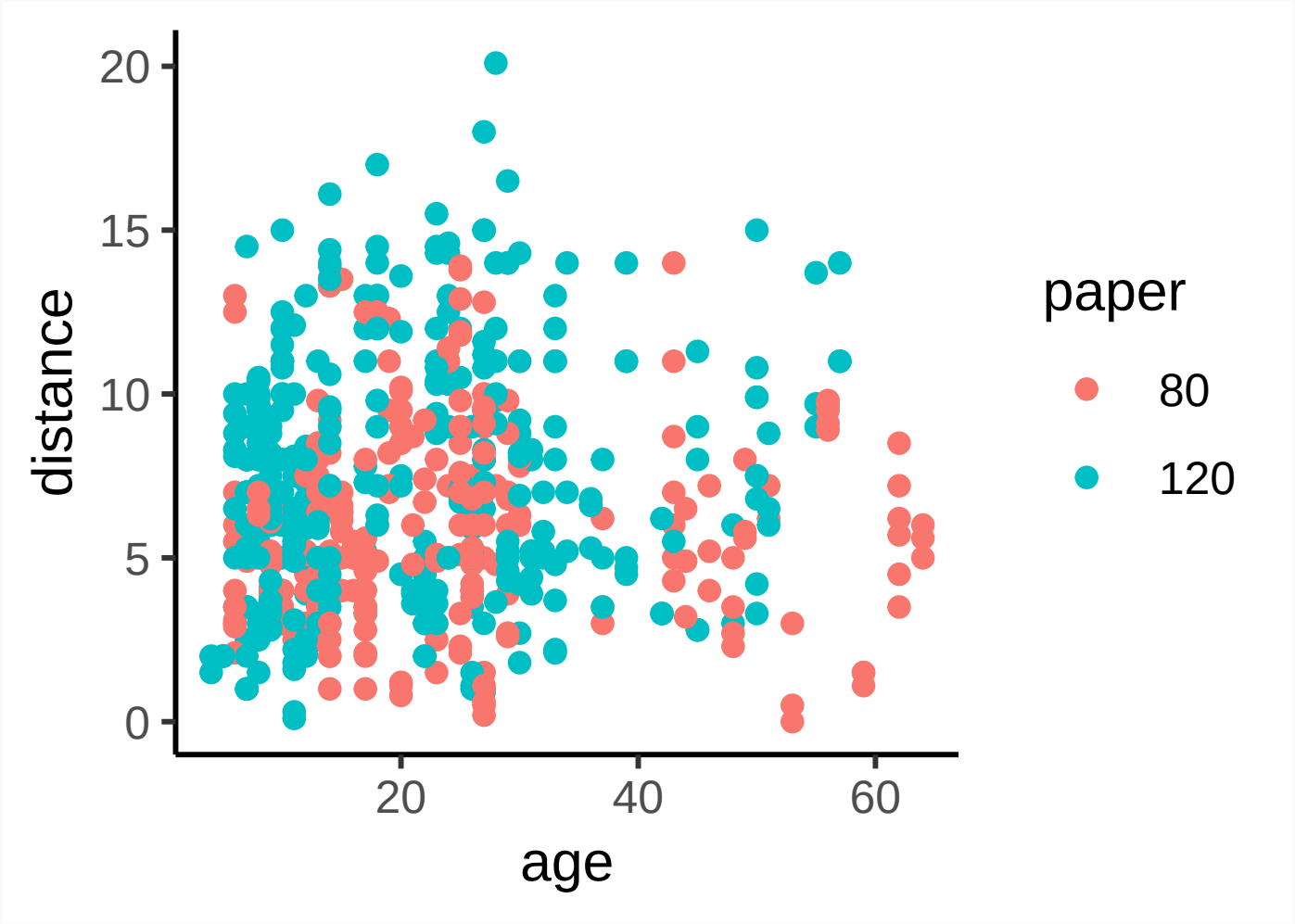
# myplot

```
myplot <- ggplot(paperplanes) +
  aes(x = age, y = distance, colour = paper) +
  geom_point()
```



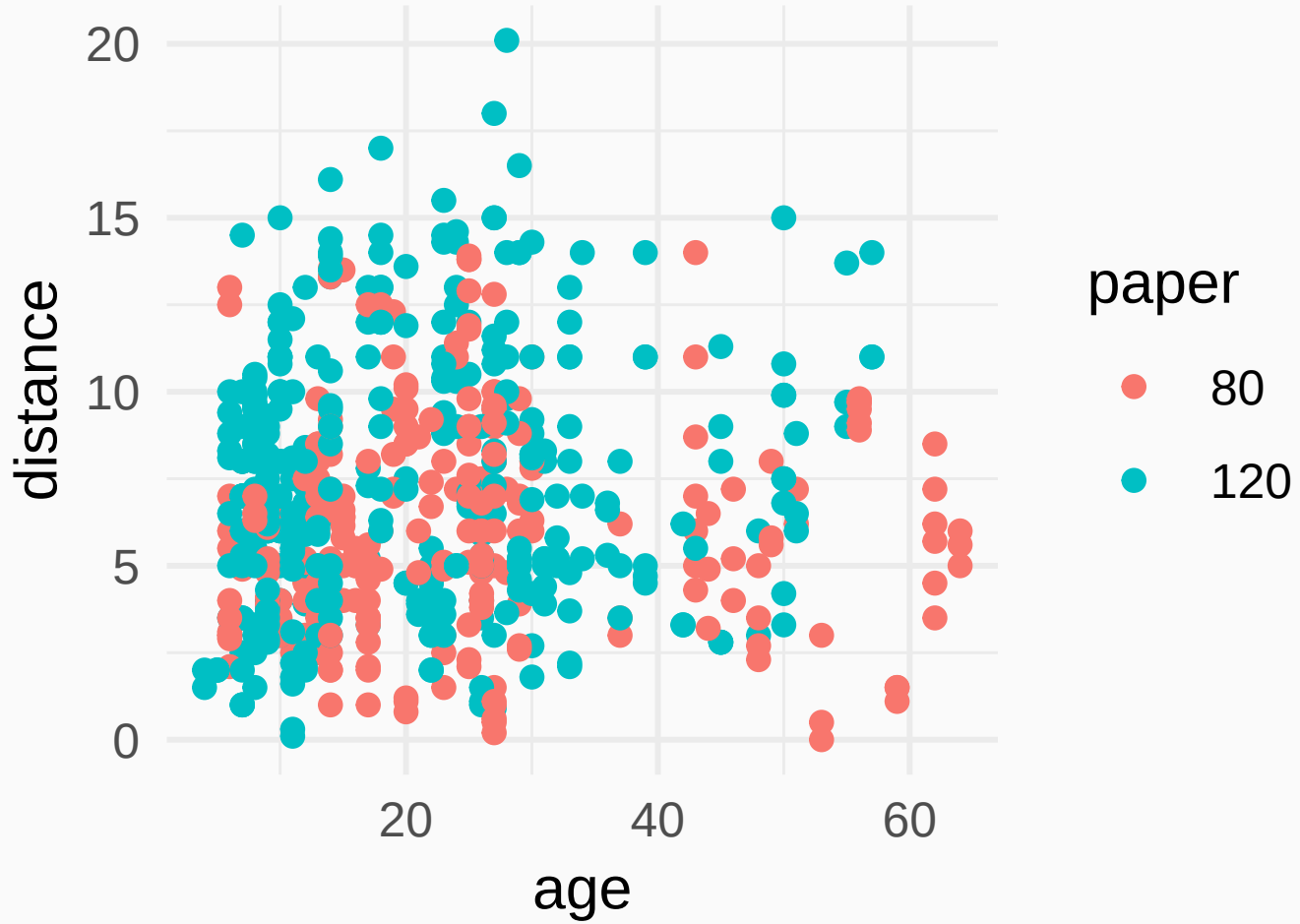
# theme_classic

```
myplot + theme_classic()
```



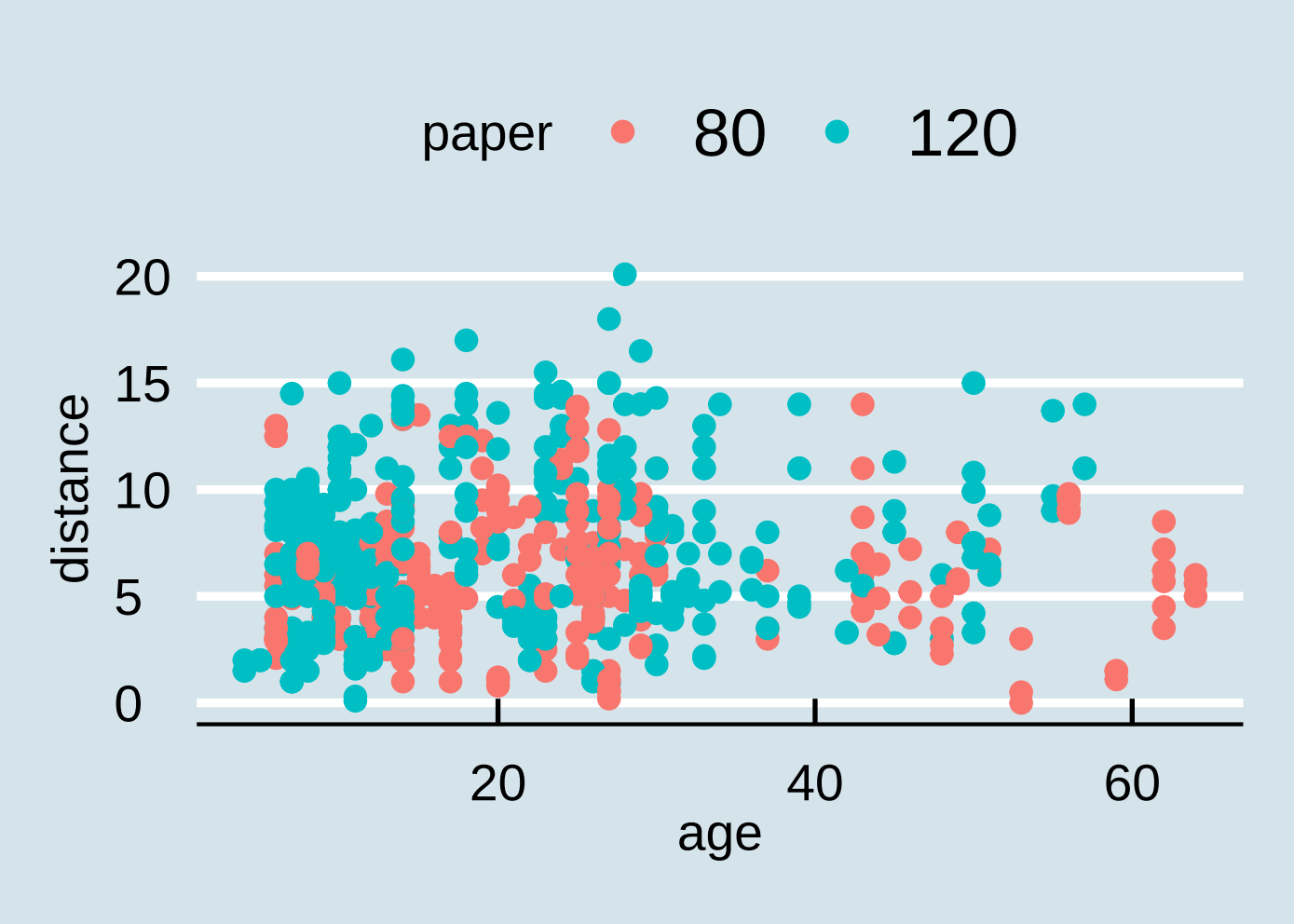
# theme_minimal

```
myplot + theme_minimal()
```



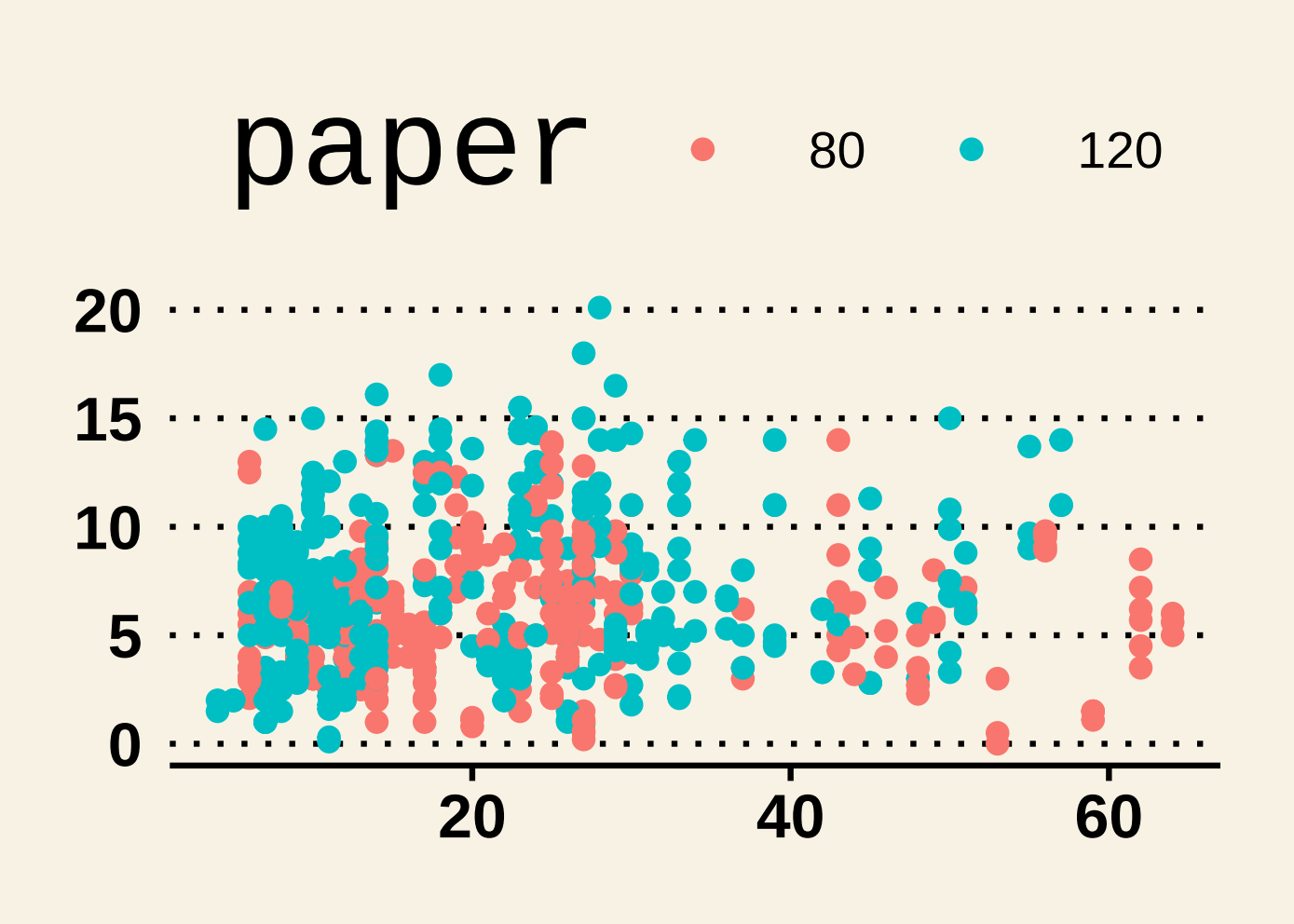
# Lots of themes out there

```
library(ggthemes)
myplot + theme_economist()
```



# Lots of themes out there

```
myplot + theme_wsj()
```

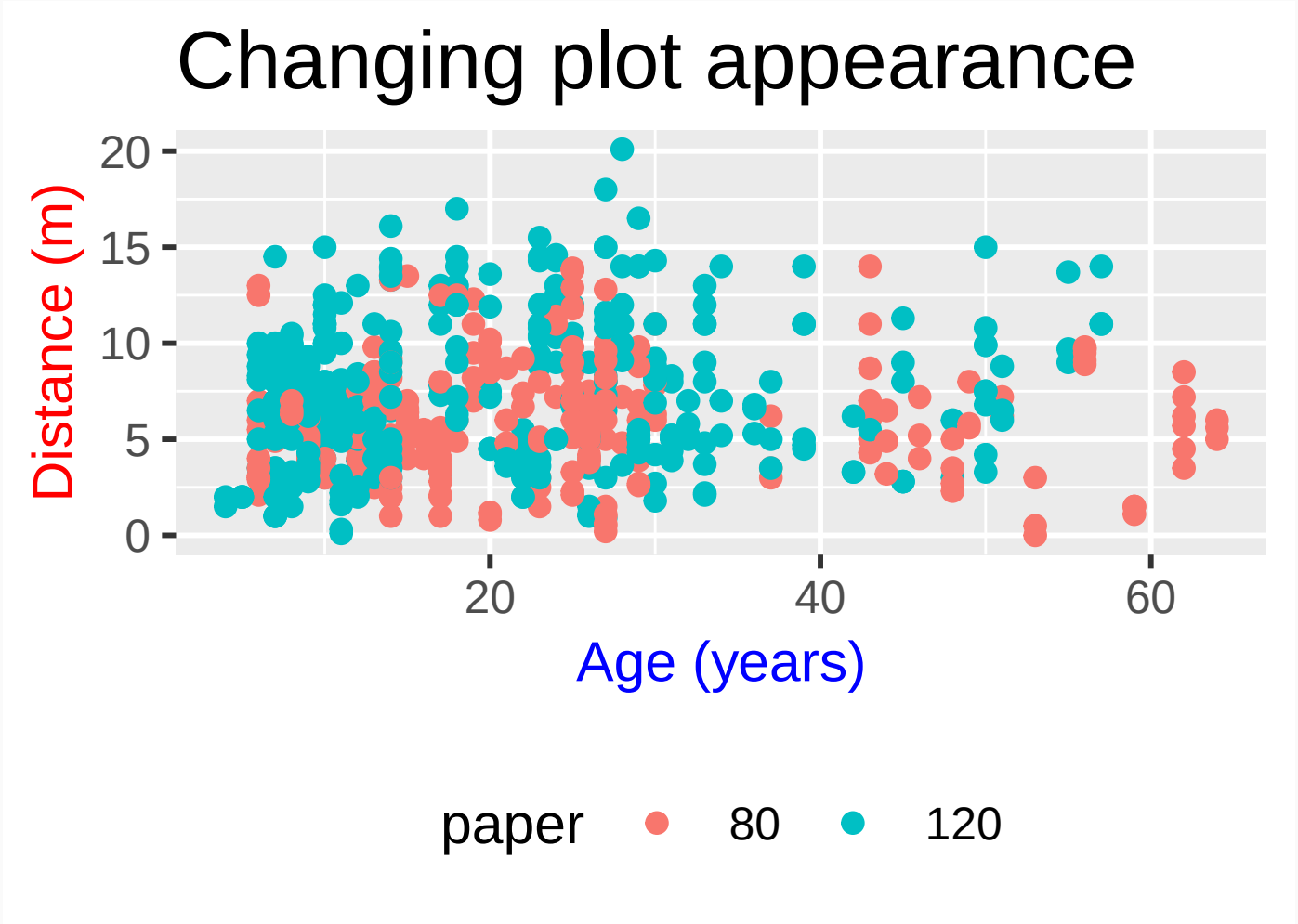


# Editing themes

```
?theme
```

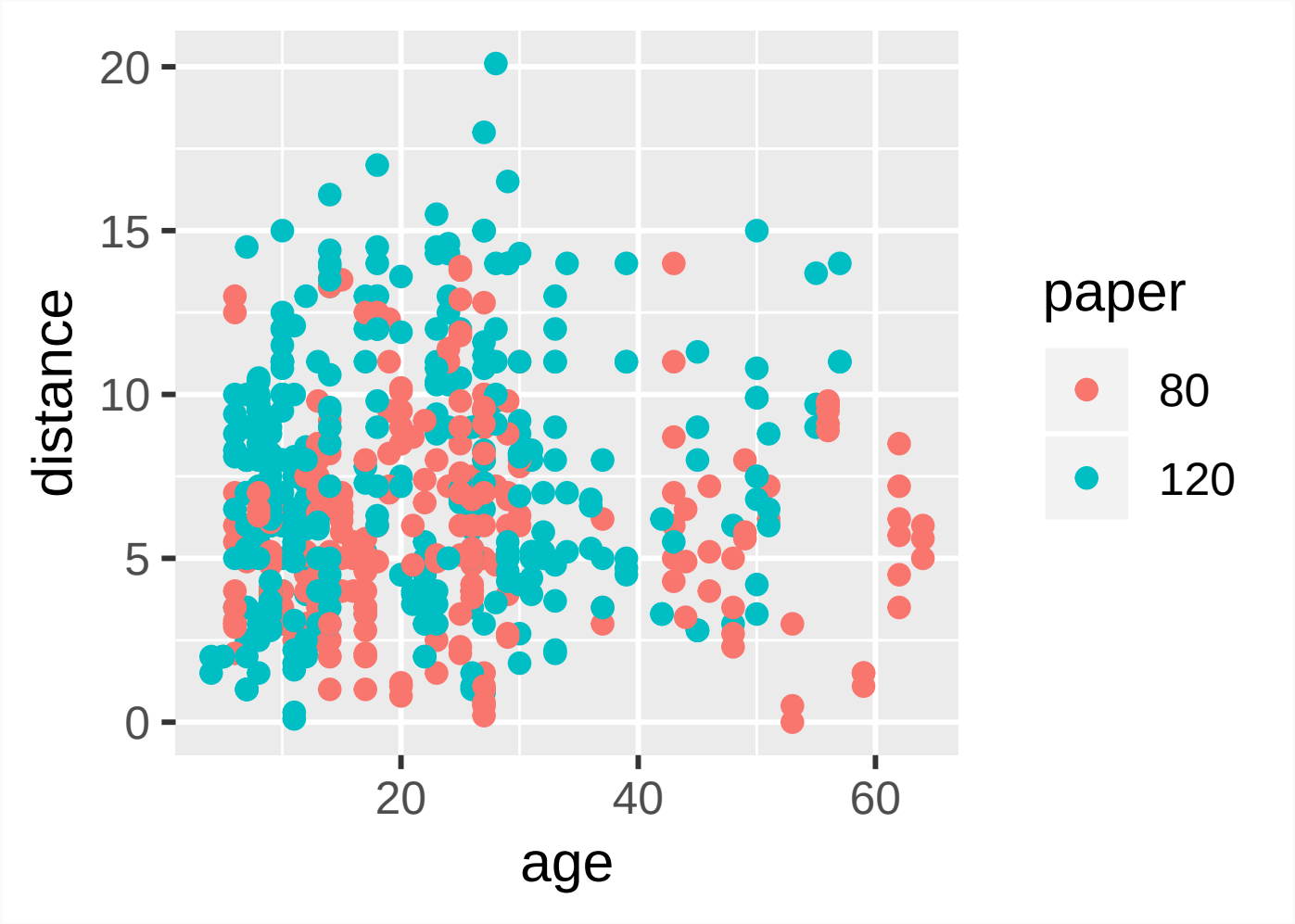
- `element_blank`
- `element_text`
- `element_line`
- `element_rect` (borders & backgrounds)

# Exercise: make a plot like this one



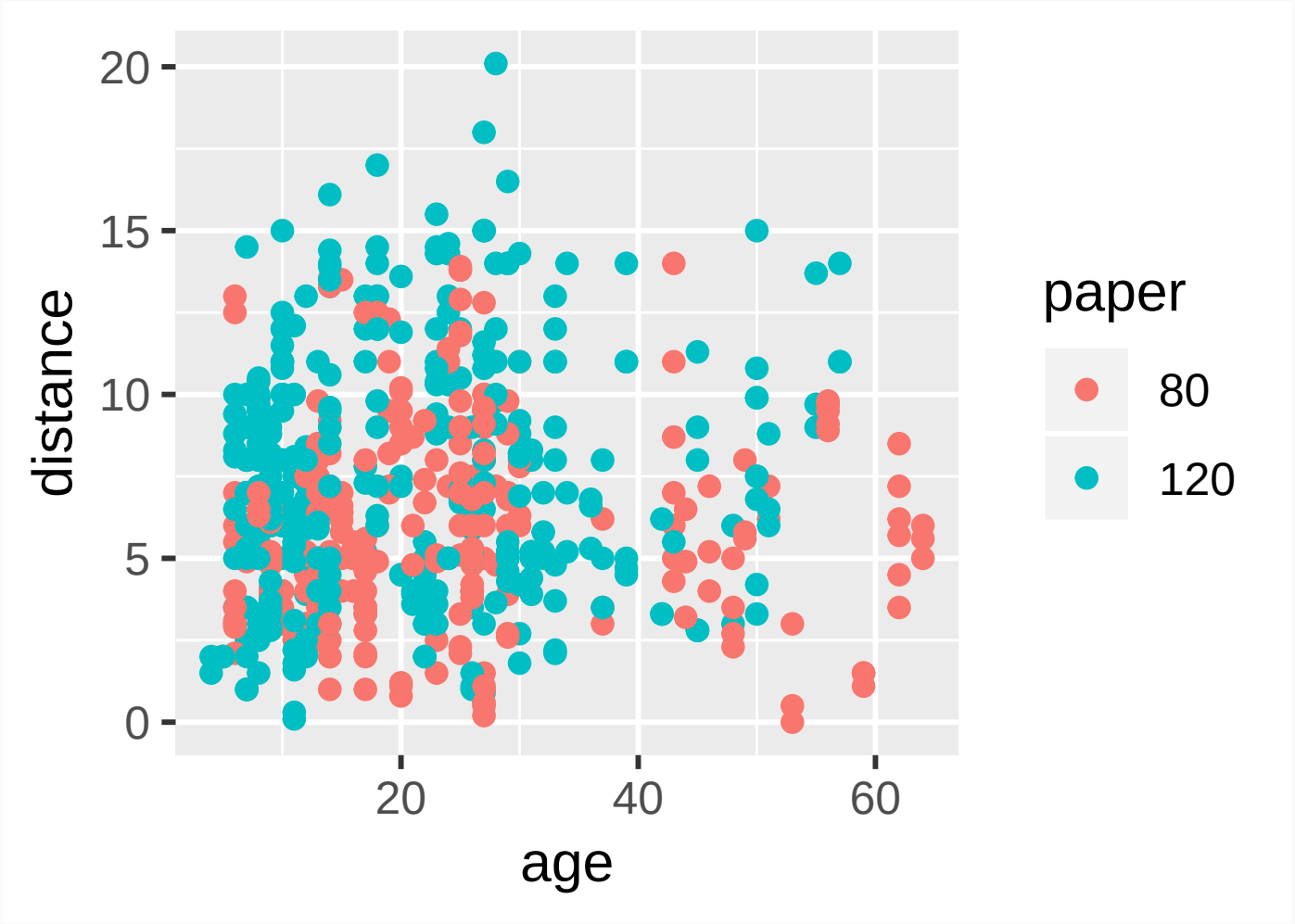
# Easily changing appearance with ggthemeassist (Rstudio addin)

https://github.com/calligross/ggthemeassist

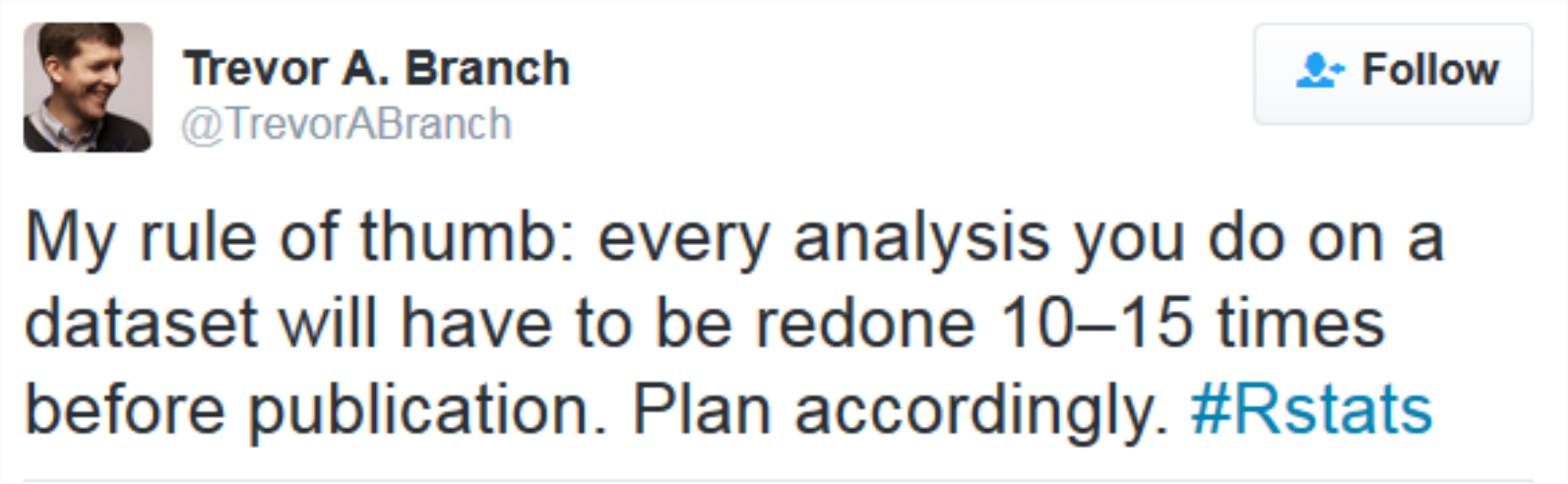


# Easily changing appearance with ggedit

https://github.com/metrumresearchgroup/ggedit



# Think twice before editing plots out of R



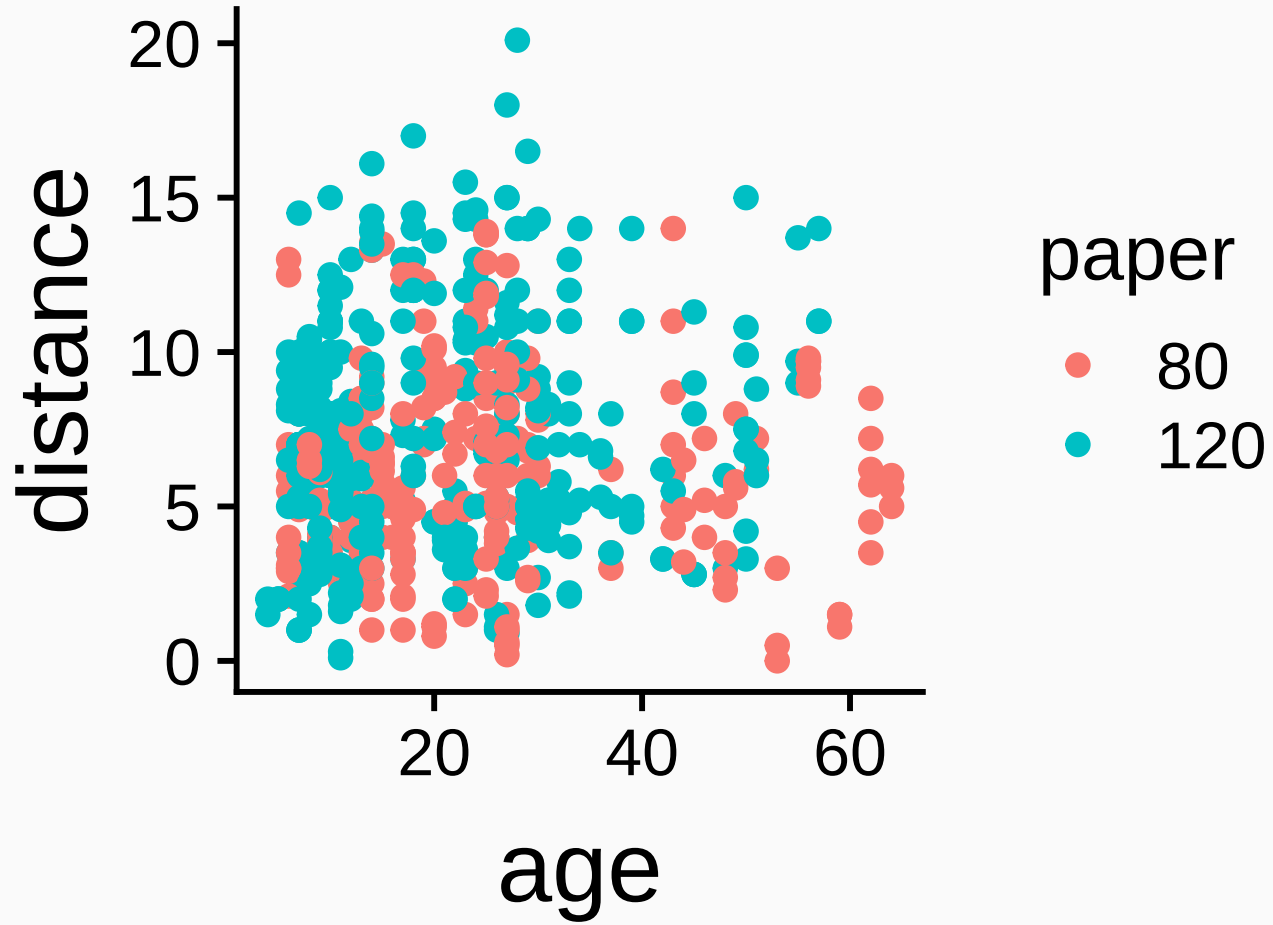

http://mbjoseph.github.io/2015/02/26/plotting.html

serialmentor.com/dataviz/choosing-the-right-visualization-software.html

# Think twice before editing plots out of R

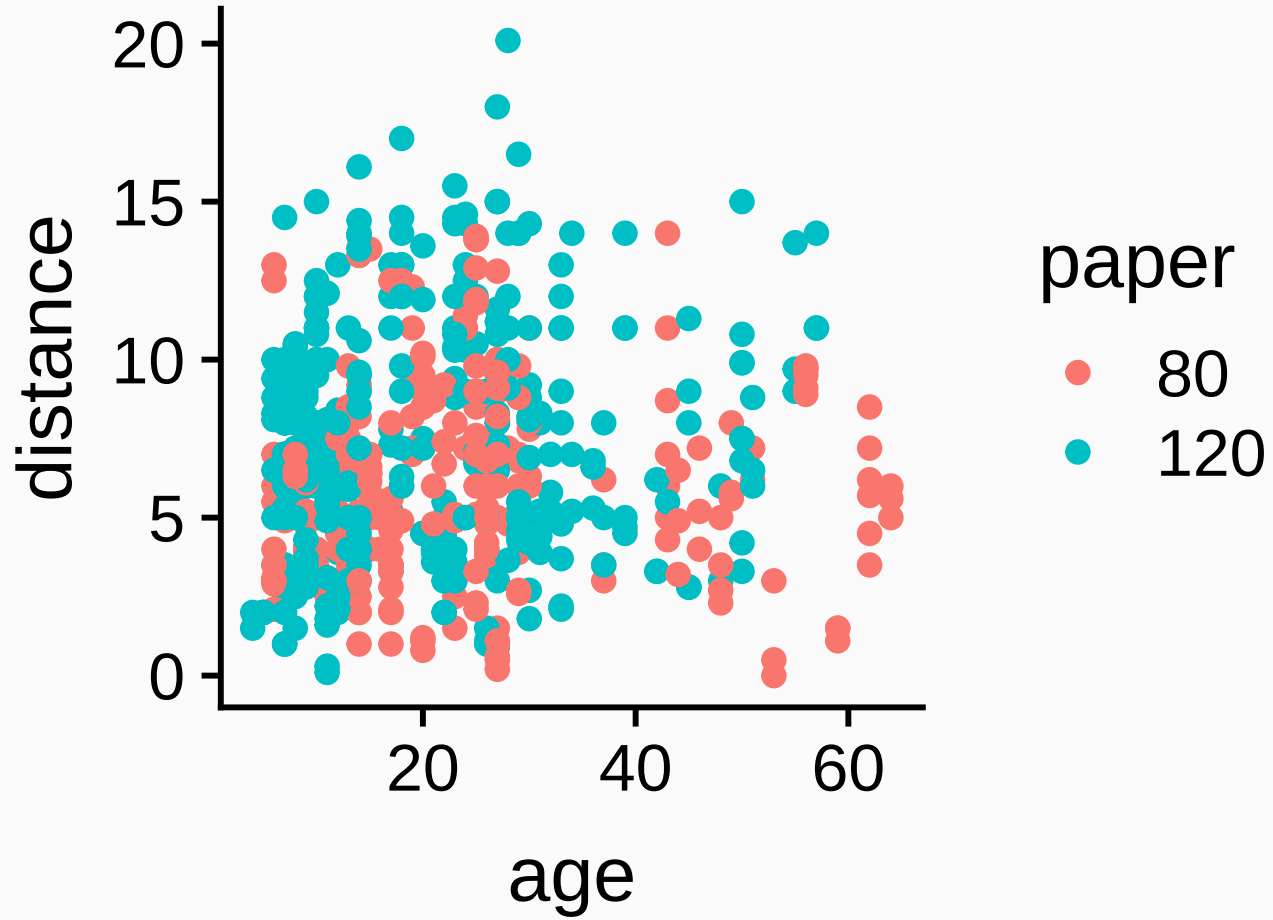
Referee #3: "Please increase font size in all figures"

```
myplot +
  theme(axis.title = element_text(size = 18))
```



# Publication-quality plots

```
library(cowplot)
myplot
```



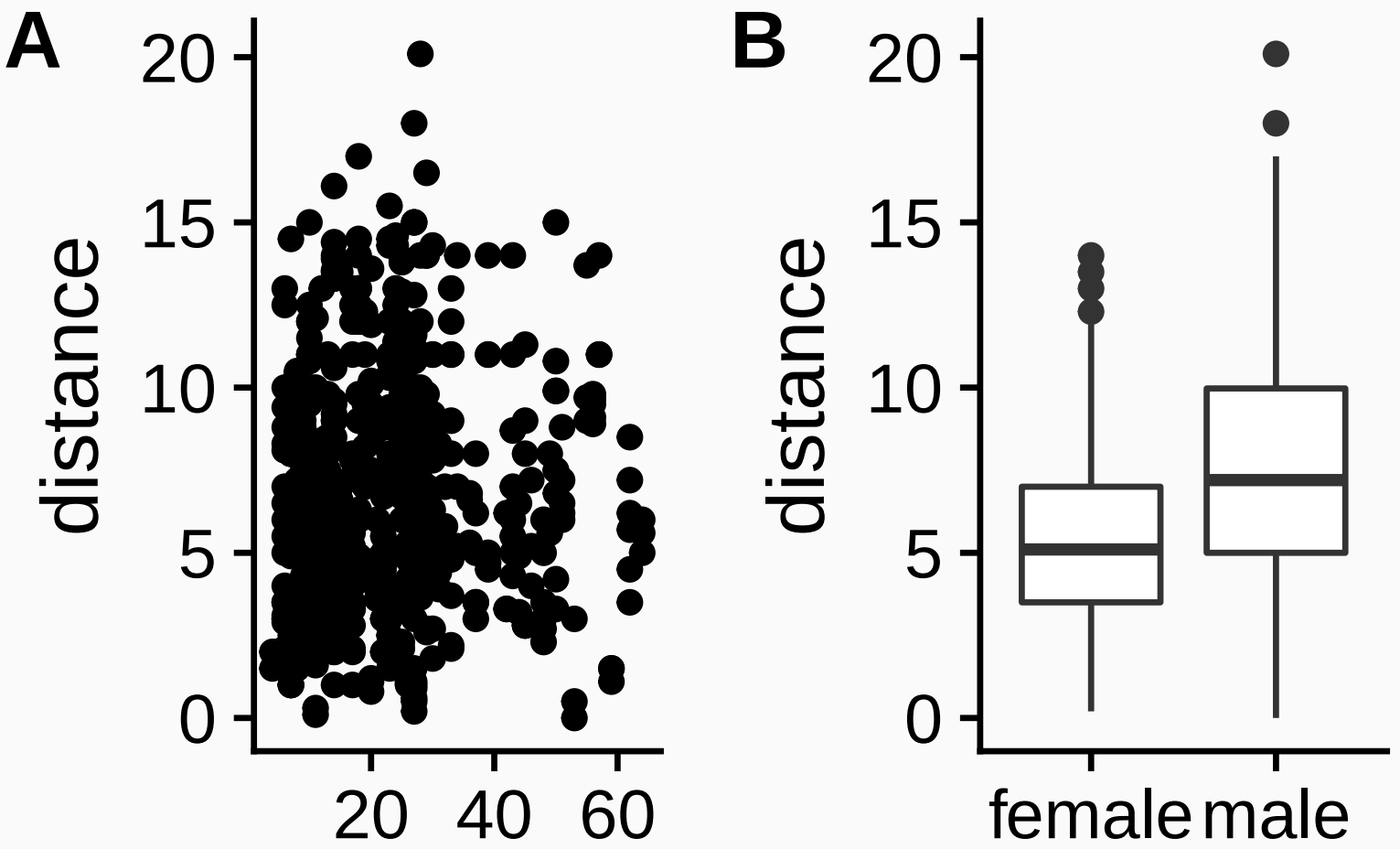
Some publication themes:

https://gist.github.com/Pakillo/c2c7ea11c528cc2ee20f#themes

# Composite figures

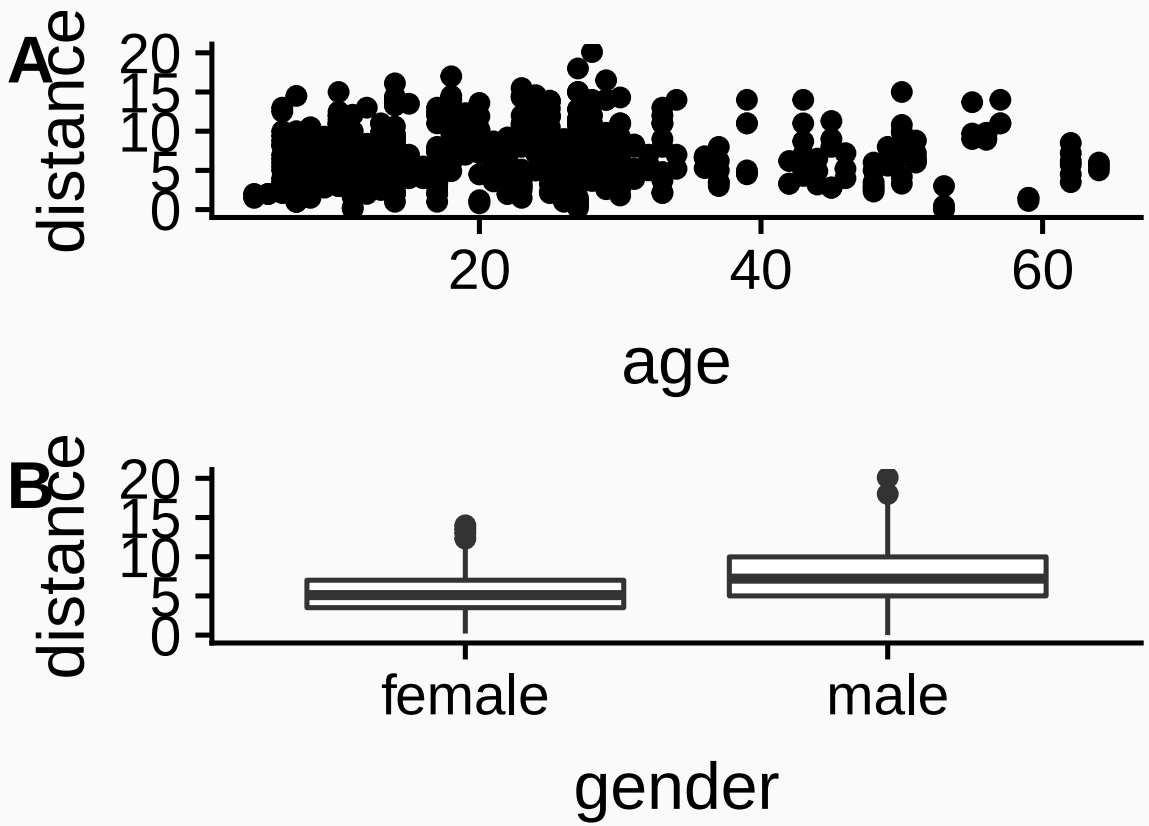
# Composite figures: cowplot

```
library(cowplot)
plot1 <- ggplot(paperplanes) + aes(age, distance) + geom_point()
plot2 <- ggplot(paperplanes) + aes(gender, distance) + geom_boxplot()
plot_grid(plot1, plot2, labels = "AUTO")
```



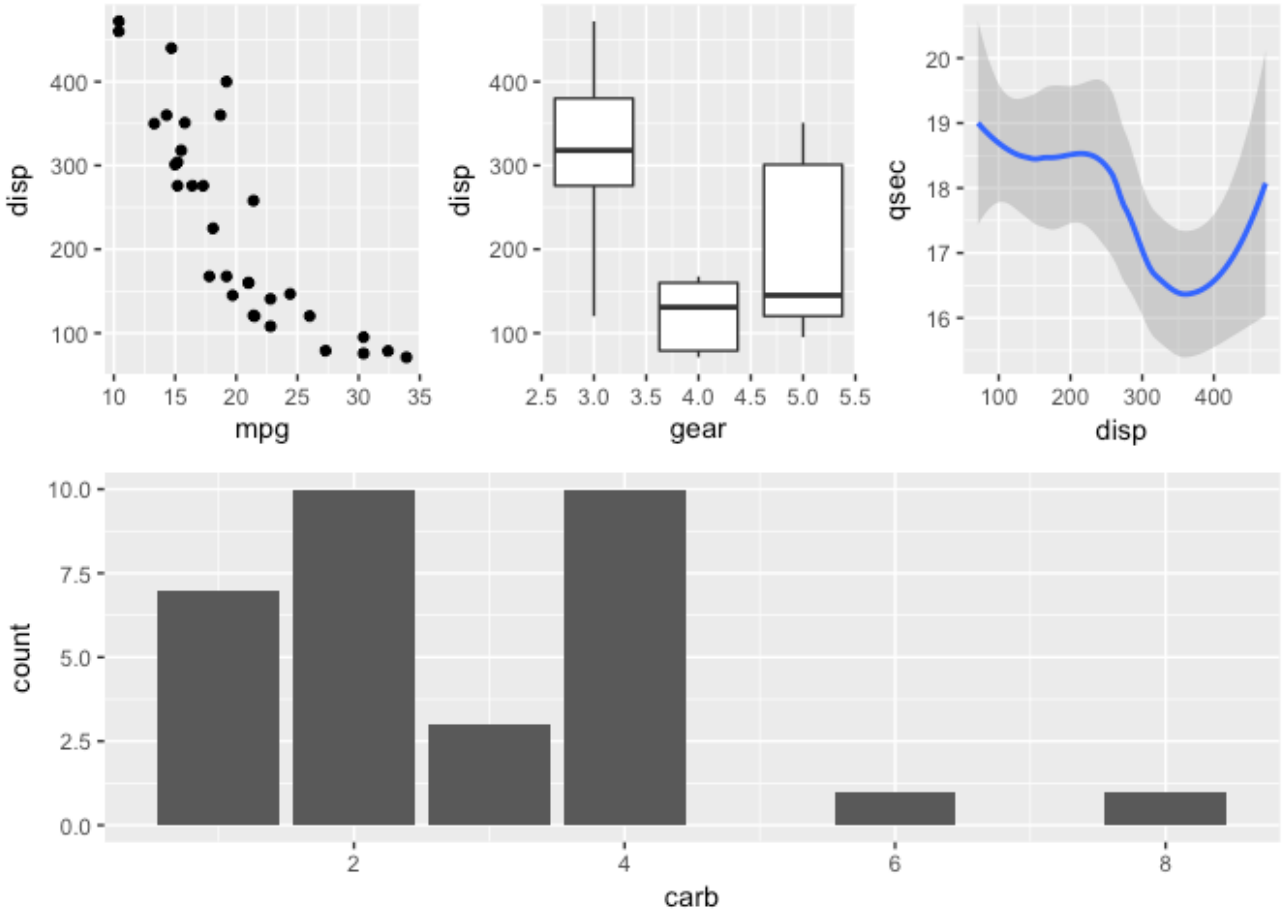
# Composite figures

```
plot_grid(plot1, plot2, labels = "AUTO", ncol = 1)
```



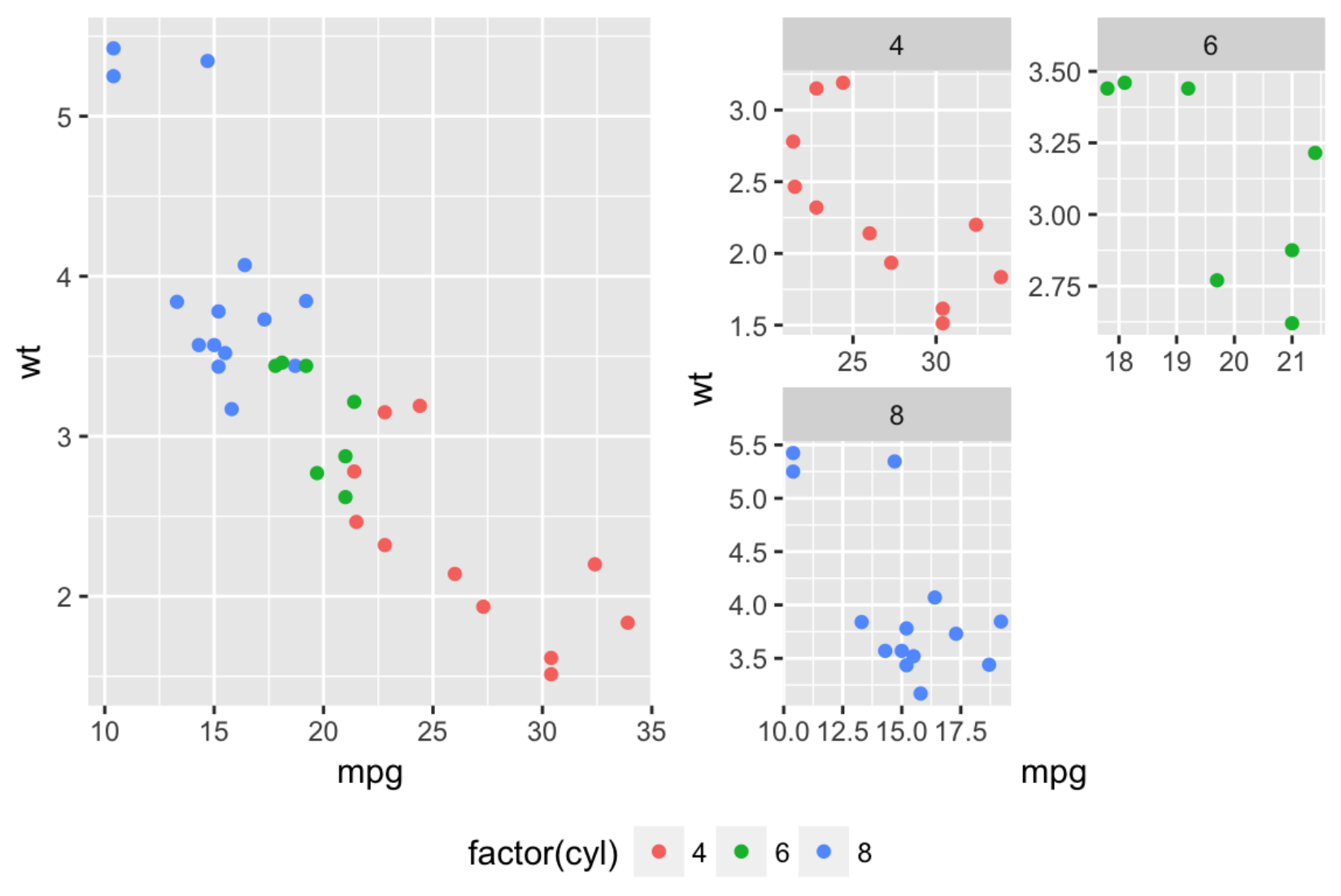
# Composite figures: patchwork

```
(p1 | p2 | p3) /
      p4
```




https://github.com/thomasp85/patchwork

# Composite figures: egg




https://cran.r-project.org/web/packages/egg/index.html

# Saving plot

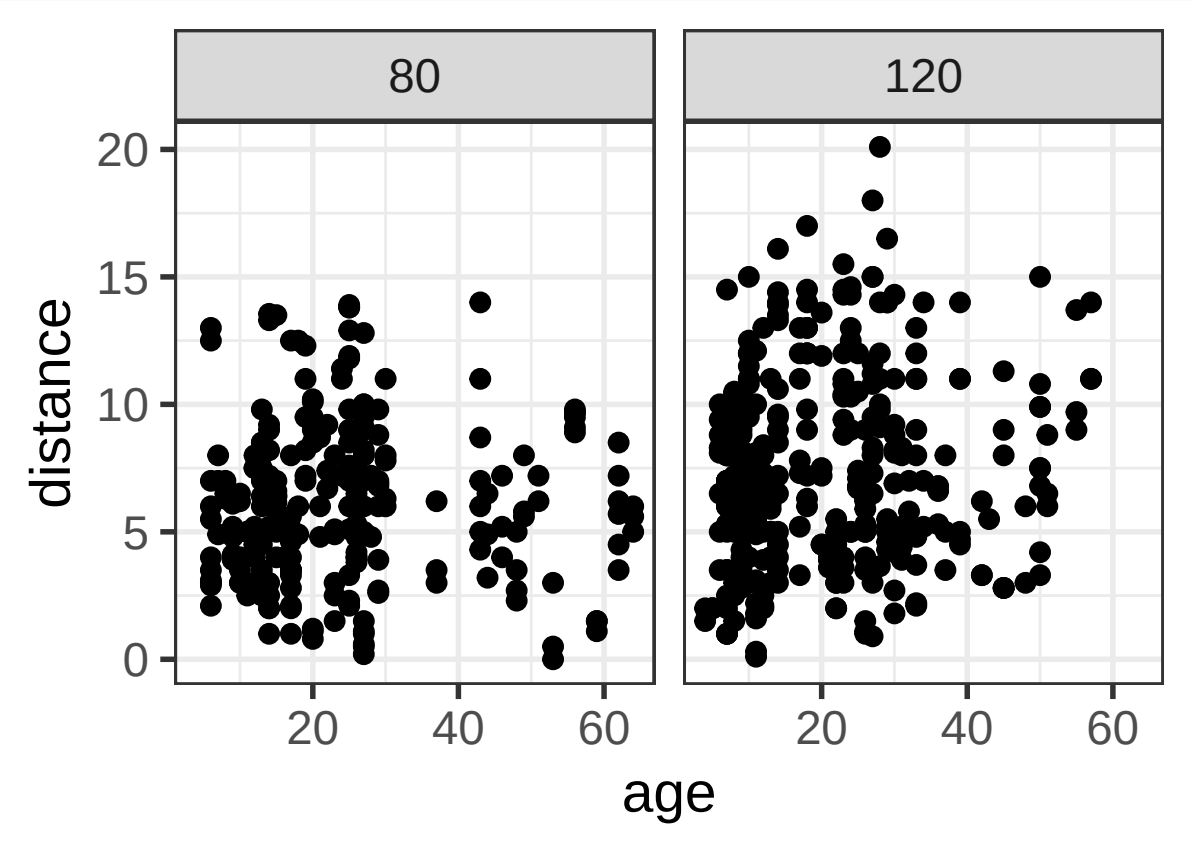
```
ggsave("myplot.pdf")
```

```
save_plot("myplot.pdf")
```

# Facetting (small multiples)

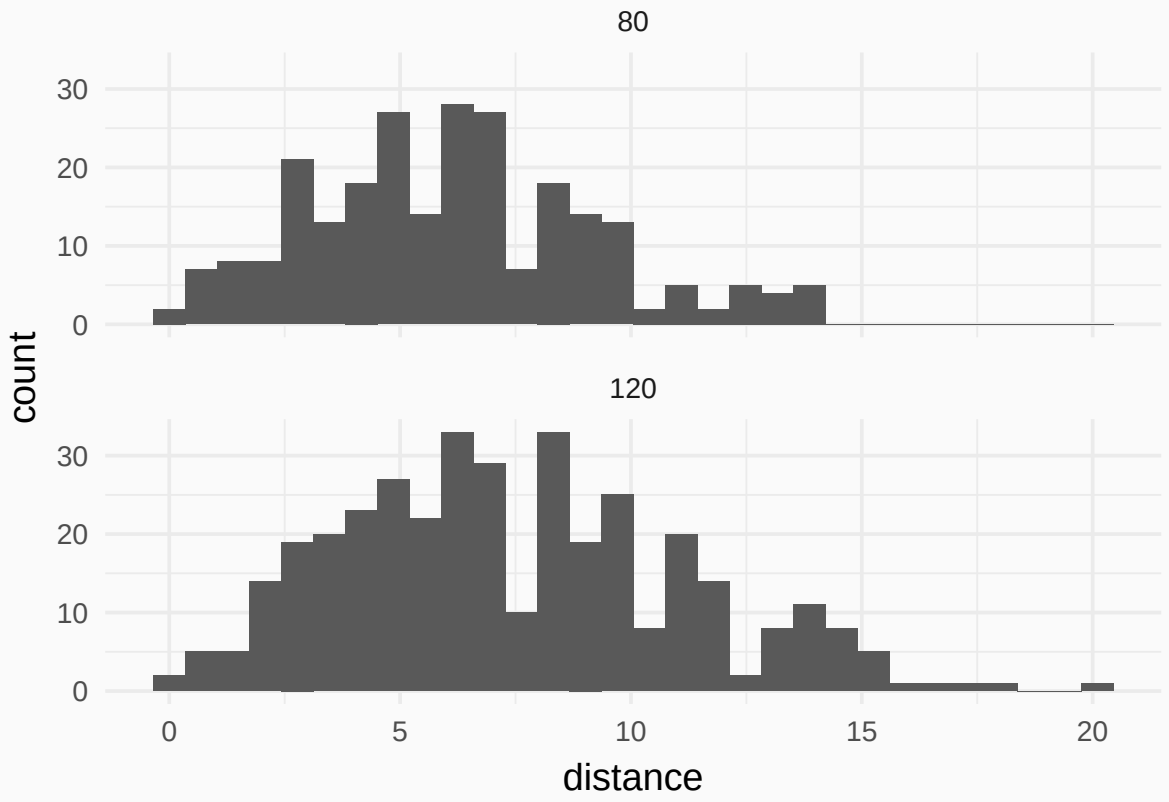
# Facetting

```
ggplot(paperplanes) + aes(age, distance) +
    geom_point() + theme_bw(base_size = 12) +
  facet_wrap(~paper)
```



# Facetting

```
ggplot(paperplanes) +
  geom_histogram(aes(distance)) +
  theme_minimal(base_size = 8) +
  facet_wrap(~paper, nrow = 2)
```

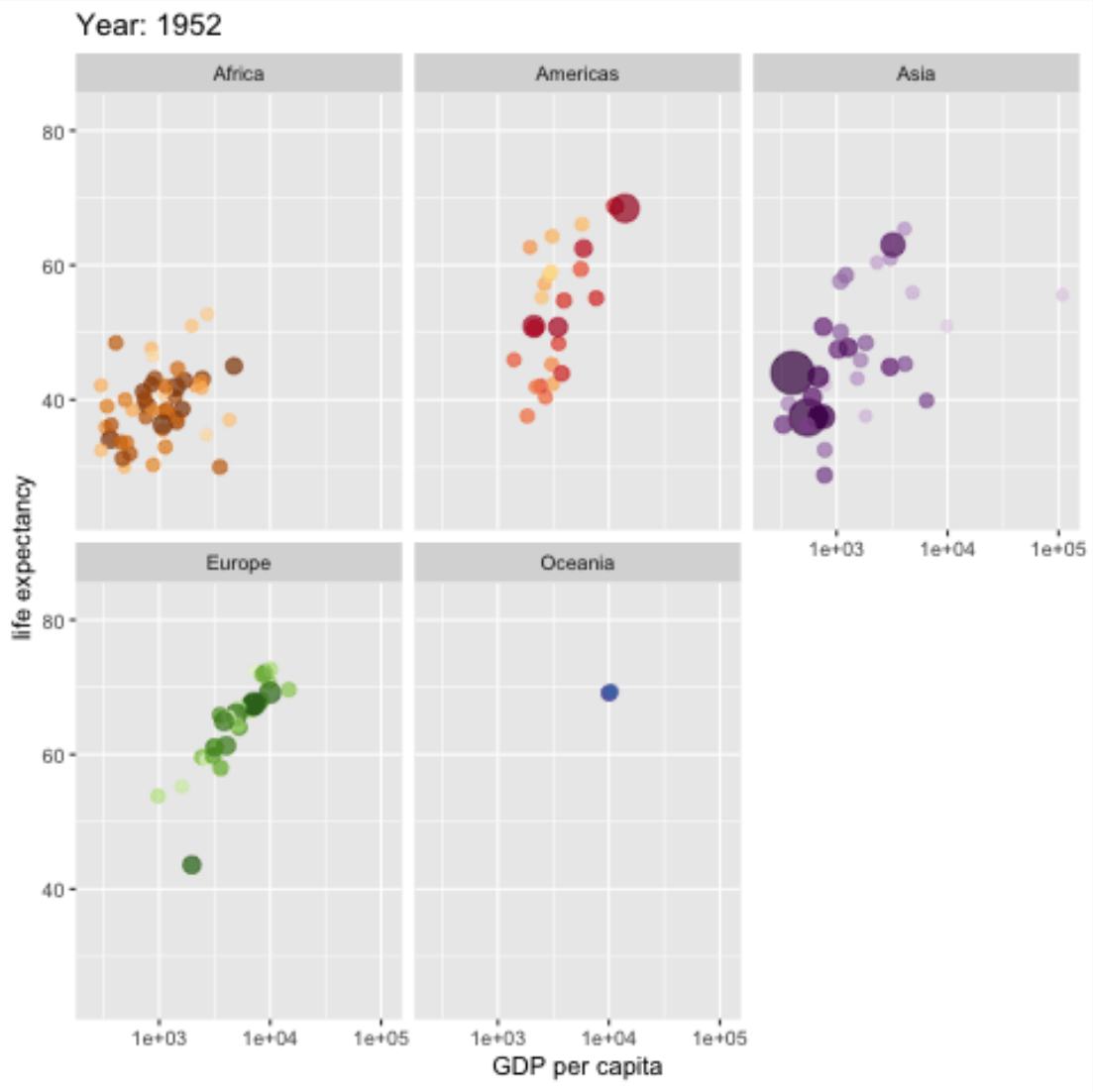


# Interactivity: plotly

```
library(plotly)
myplot <- ggplot(paperplanes) +
  aes(age, distance)) +
  geom_point()
ggplotly(myplot)
```

# Animated graphs

https://github.com/thomasp85/gganimate

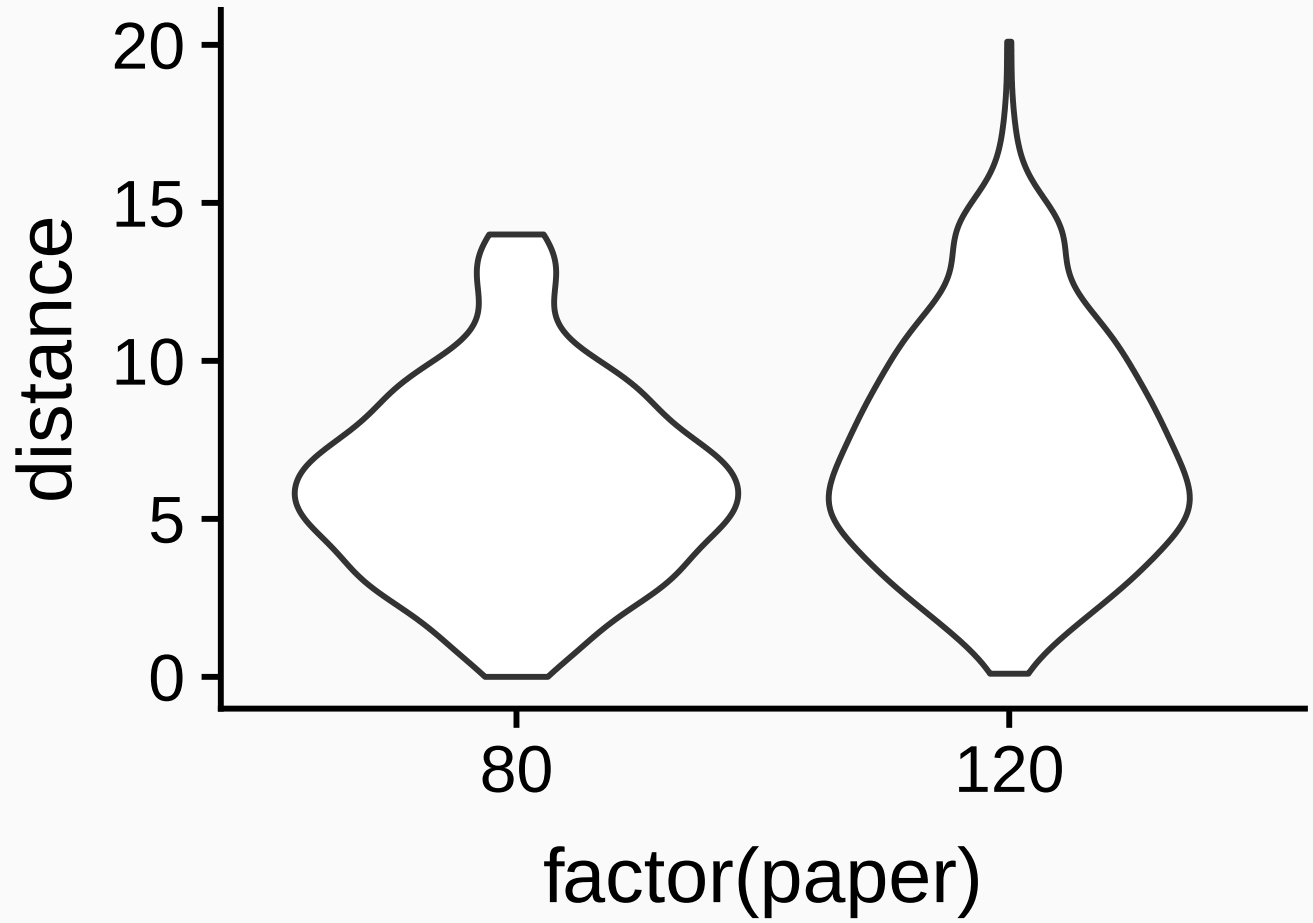


# Summary

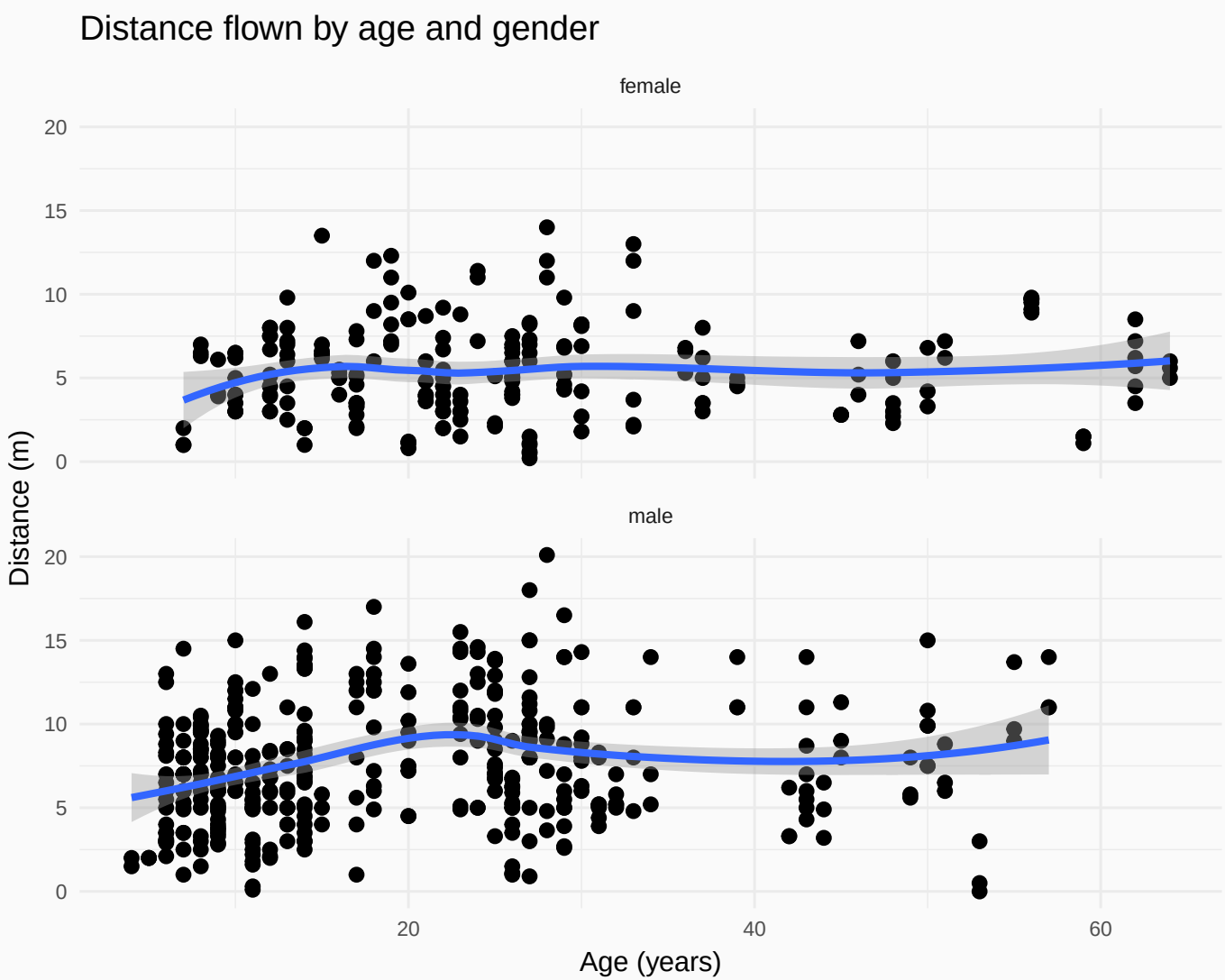
# Grammar of graphics

- **Data** (tidy data frame)
- **Layers** (*geoms*: points, lines, polygons. . . )
- **Aesthetics** mappings (x, y, size, colour. . . )
- **Scales** (colour, size, shape. . . )
- **Facets** (small multiples)
- **Themes** (appearance)
- **Coordinate system** (Cartesian, polar, map projections. . . )

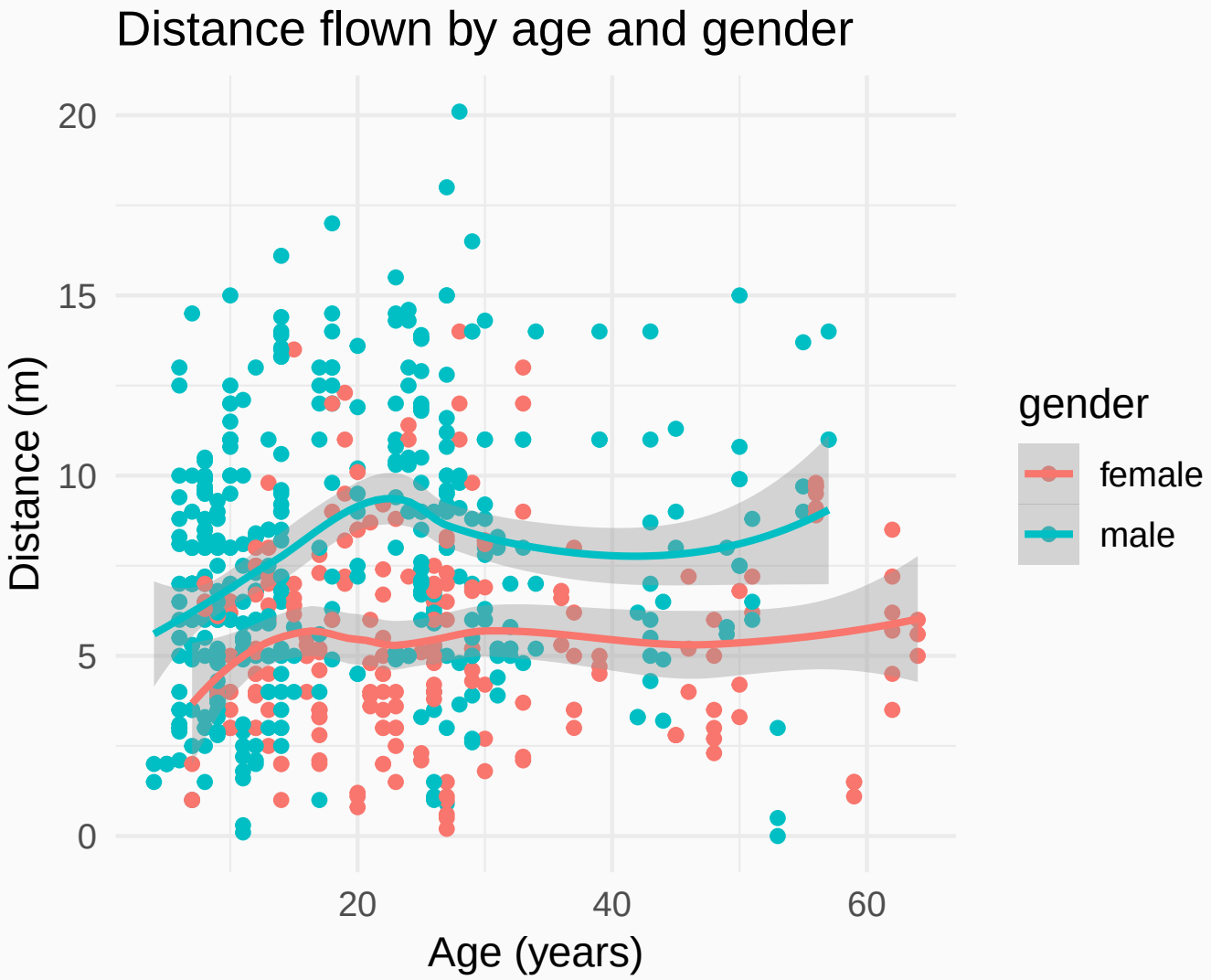
# Exercise: make a plot like this one



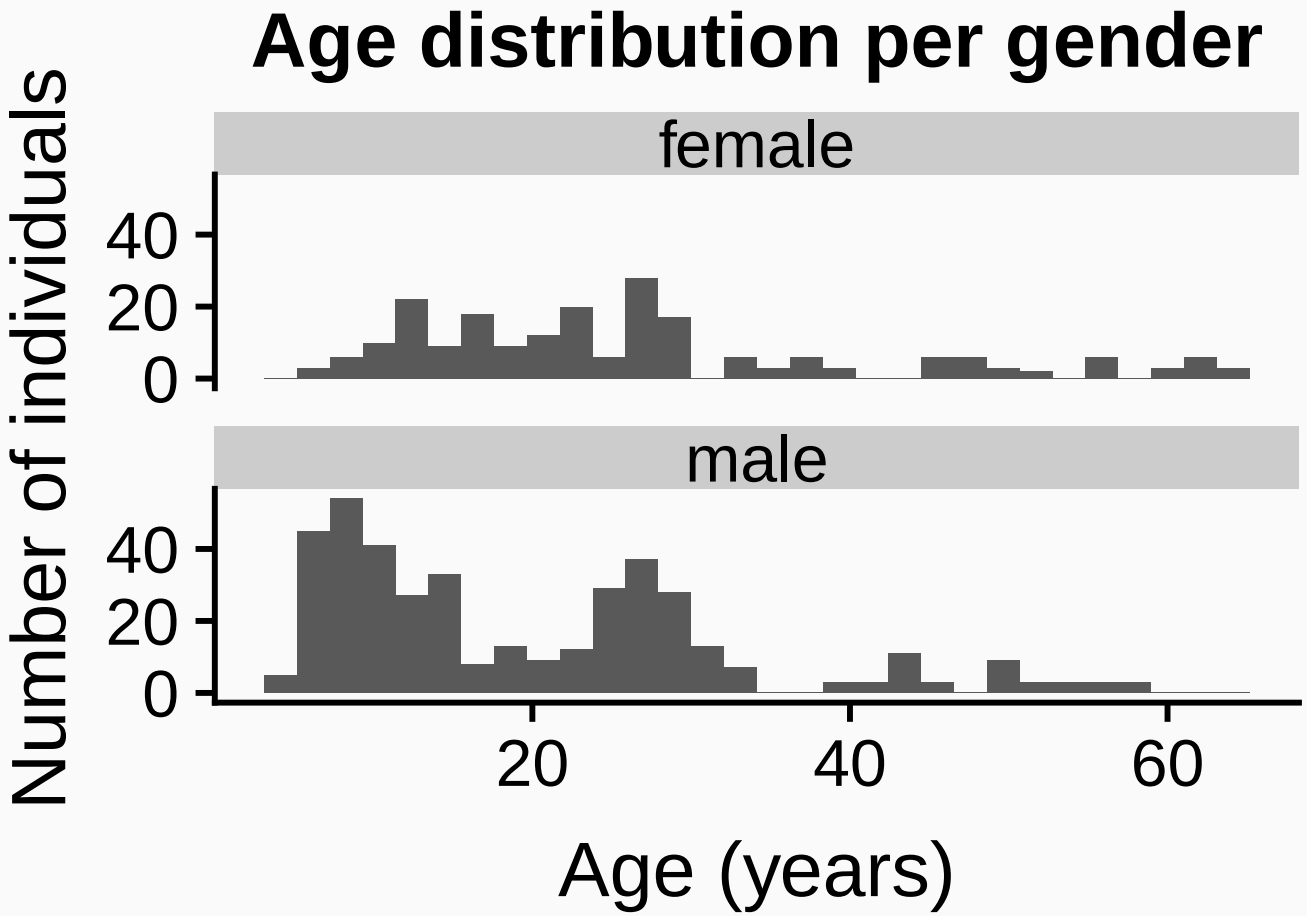
# Exercise: make a plot like this one



# Exercise: make a plot like this one

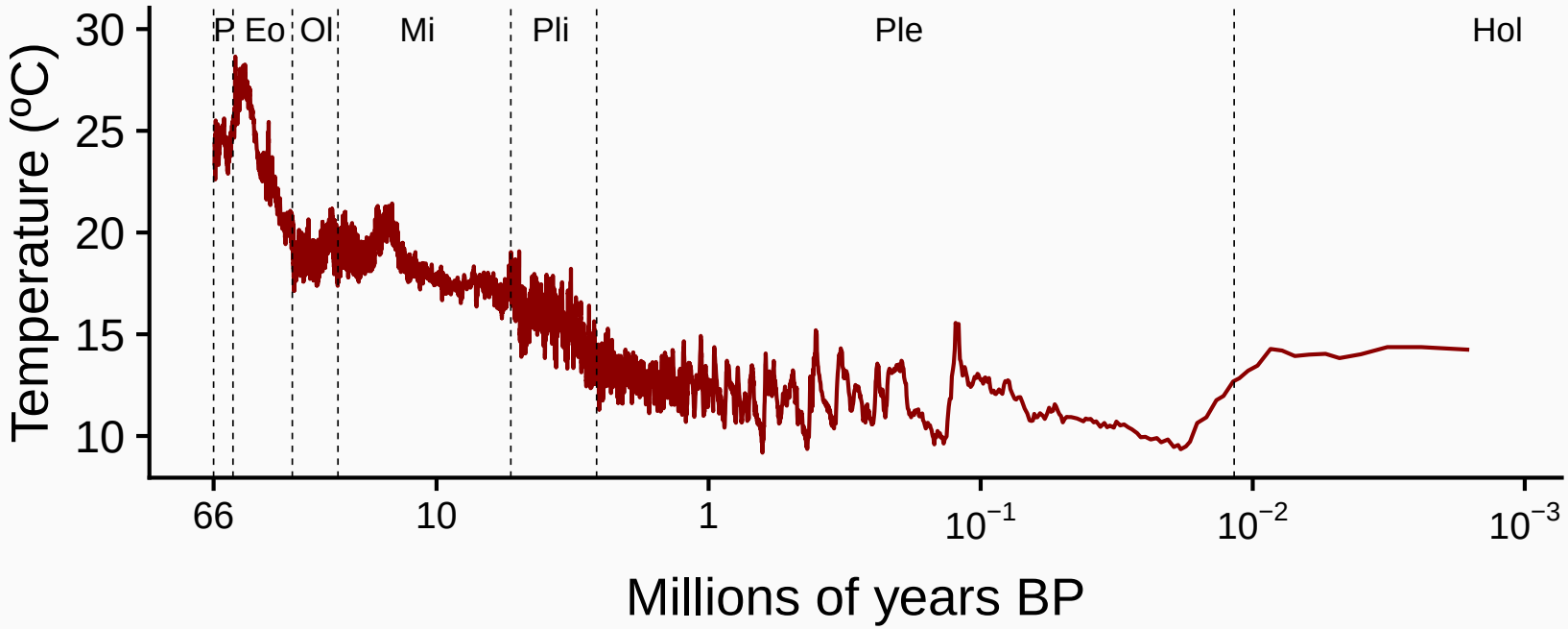


## Exercise: make a plot like this one

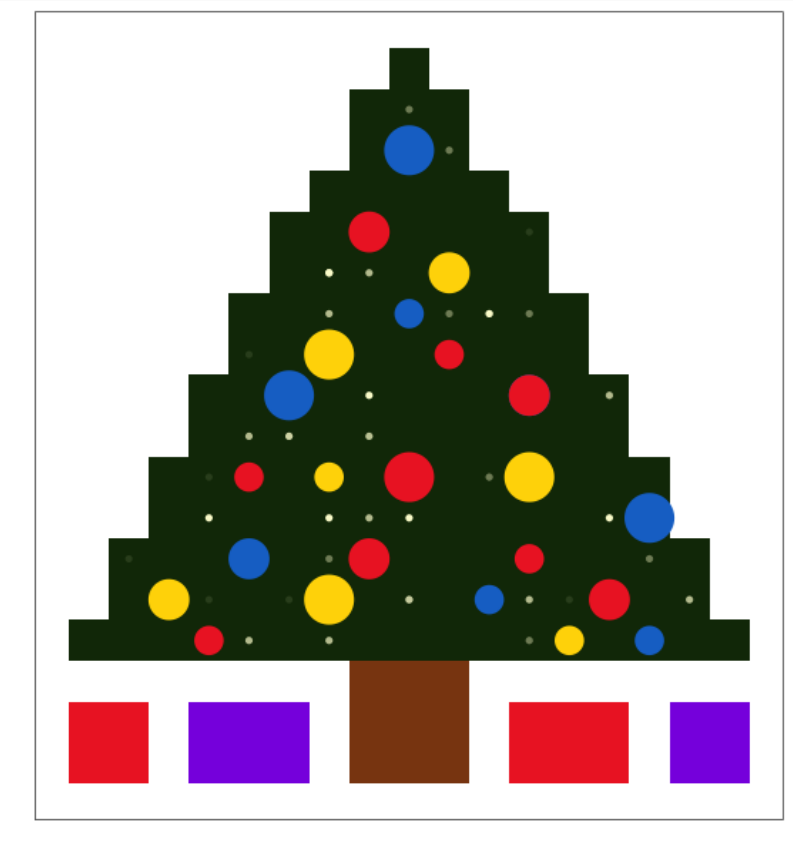


# Exercise: make a plot like this one

Data from http://www.columbia.edu/~mhs119/Sensitivity+SL+CO2/Table.txt



# Exercise: make a plot like this one



# END


Slides and source code available at https://github.com/Pakillo/ggplot-intro