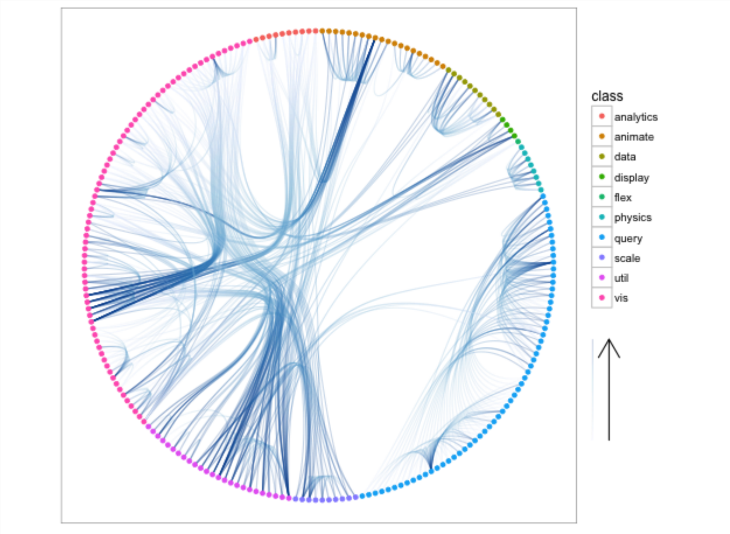
Data visualisation with ggplot2

Francisco Rodriguez-Sanchez (@frod_san)

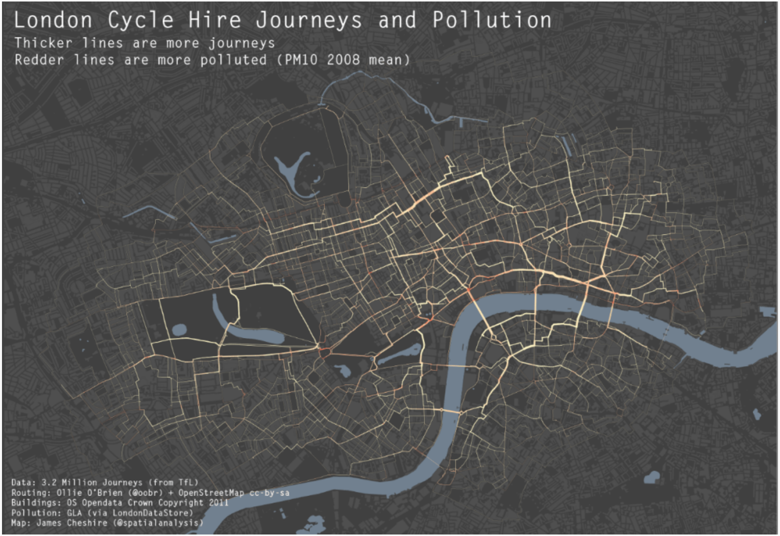
November 2016

Made with ggplot



https://github.com/thomasp85/ggraph

Made with ggplot



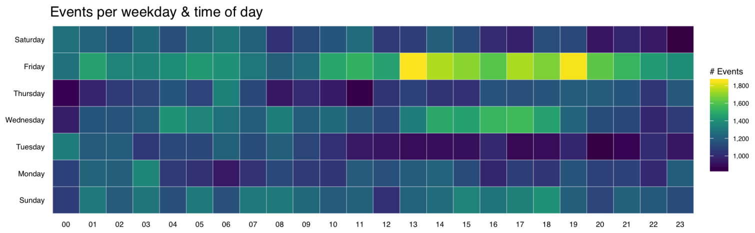
London Cycle Hire Journeys and Pollution

Thicker lines are more journeys
Redder lines are more polluted (PM10 2008 mean)

Data: 3.2 Million Journeys (from TfL)
Routing: Ollie O'Brien (@oobr) + OpenStreetMap cc-by-sa
Buildings: OS Opendata Crown Copyright 2011
Pollution: GLA (via LondonDataStore)
Map: James Cheshire (@spatialanalysis)

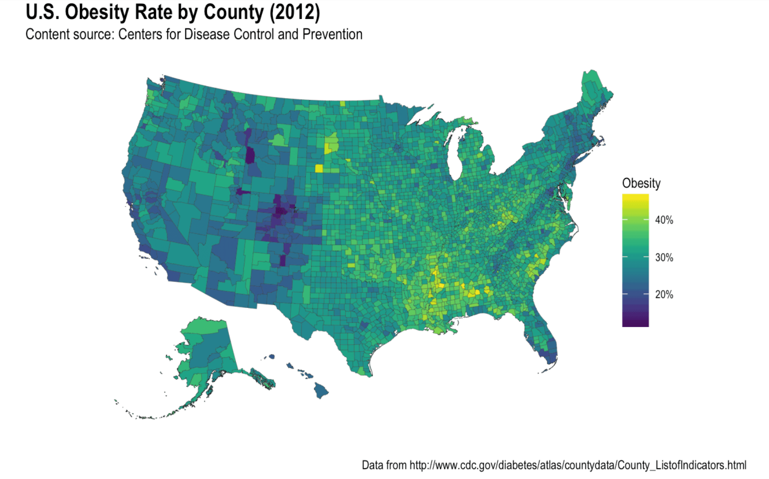
http://spatial.ly/2012/02/great-maps-ggplot2/

Made with ggplot



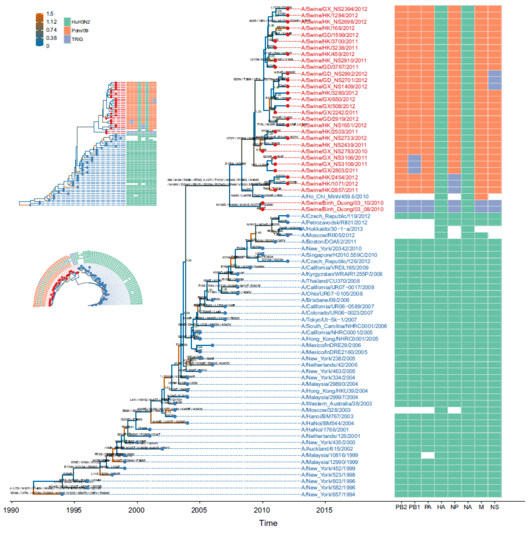
https://rud.is/b/2016/02/14/making-faceted-heatmaps-with-ggplot2/

Made with ggplot


U.S. Obesity Rate by County (2012)
Content source: Centers for Disease Control and Prevention
Obesity
40%
30%
20%
Data from http://www.cdc.gov/diabetes/atlas/countydata/County_ListofIndicators.html

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

Made with ggplot

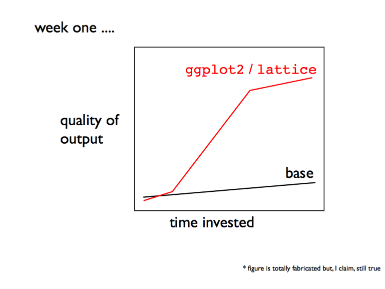


https://guangchuangyu.github.io/ggtree/

Why ggplot

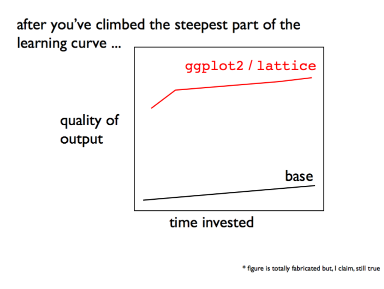
- Extremely powerful and flexible
- Consistent (grammar of graphics)
- Very powerful user base and active development

At the beginnning it’s hard, but then it pays off



week one ....

* figure is totally fabricated but, I claim, still true



after you’ve climbed the steepest part of the learning curve ...

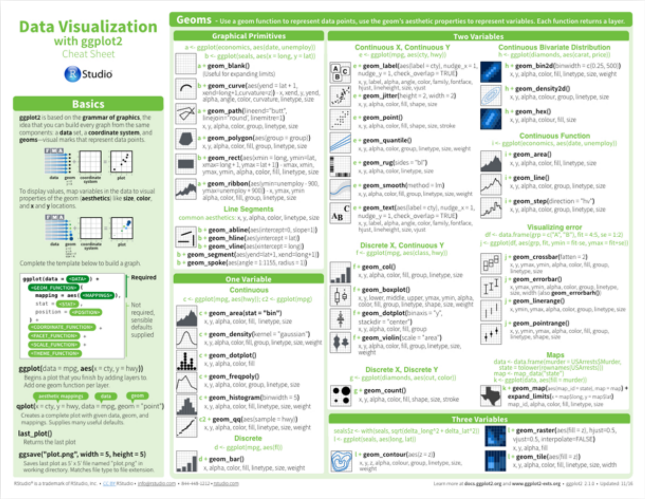
* figure is totally fabricated but, I claim, still true

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

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.

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

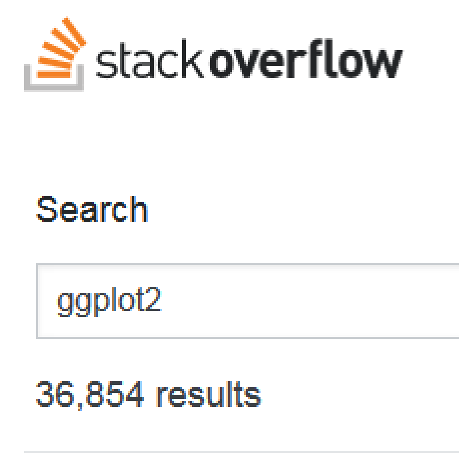
Learn more at **docs.ggplot2.org** and **www.ggplot2-exts.org** • ggplot2 2.1.0 • Updated 11/16

https://www.rstudio.com/wp-content/uploads/2016/11/ggplot2-cheatsheet-2.1.pdf

Repos of figures + code

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

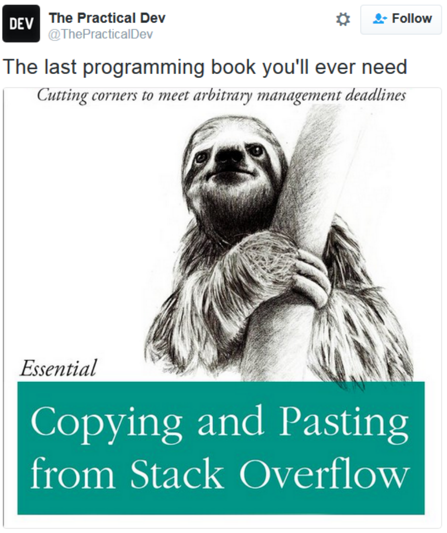
Find answers for all your questions in Stack Overflow



Search

ggplot2

36,854 results



The Practical Dev
@ThePracticalDev

The last programming book you'll ever need

Cutting corners to meet arbitrary management deadlines

Essential

Copying and Pasting from Stack Overflow

Building a ggplot

Our example dataset: tree heights and DBH

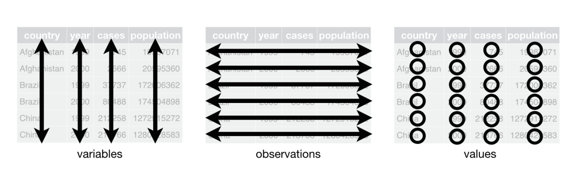
http://tinyurl.com/treesdata

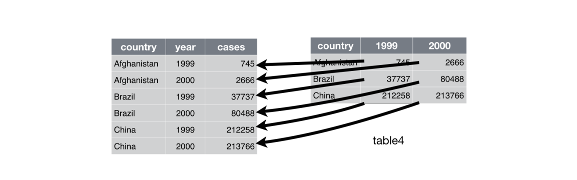
- One species
- 10 plots
- 1000 trees
- Number of trees per plot ranging from 4 to 392

```r
trees <- read.csv("data/trees.csv")
summary(trees[, 1:3])
```

```
      plot           dbh            height     
 Min.   : 1.0   Min.   : 5.06   Min.   :13.40  
 1st Qu.: 1.0   1st Qu.:17.69   1st Qu.:29.68  
 Median : 2.0   Median :28.62   Median :36.55  
 Mean   : 2.7   Mean   :27.88   Mean   :36.51  
 3rd Qu.: 4.0   3rd Qu.:38.97   3rd Qu.:43.33  
 Max.   :10.0   Max.   :49.92   Max.   :59.30  
```

Data must be a tidy data frame





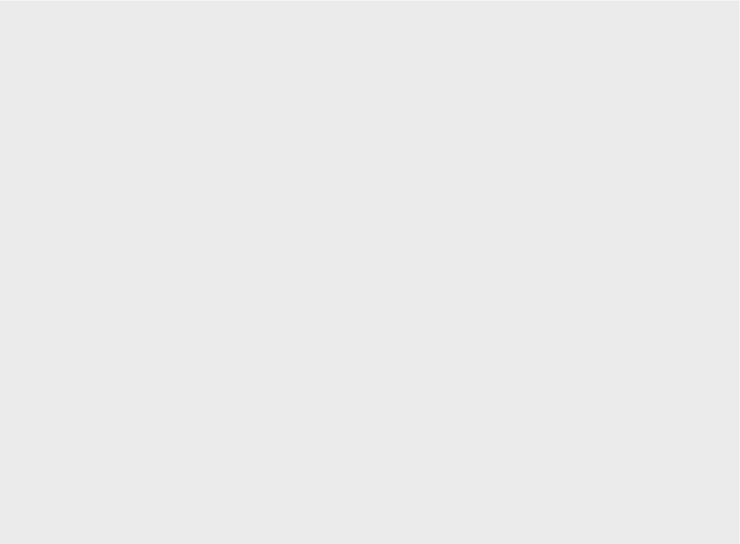
country	year	cases
Afghanistan	1999	745
Afghanistan	2000	2666
Brazil	1999	37737
Brazil	2000	80488
China	1999	212258
China	2000	213766

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

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

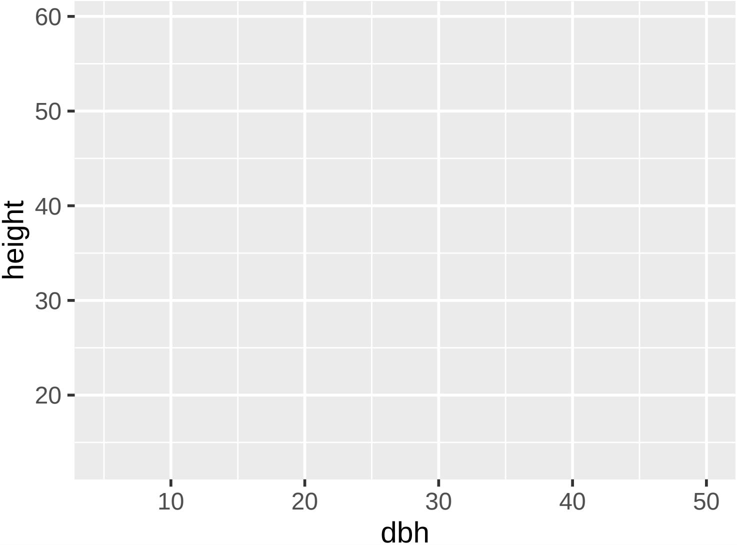
Calling ggplot

```
library(ggplot2)
ggplot(trees)
```



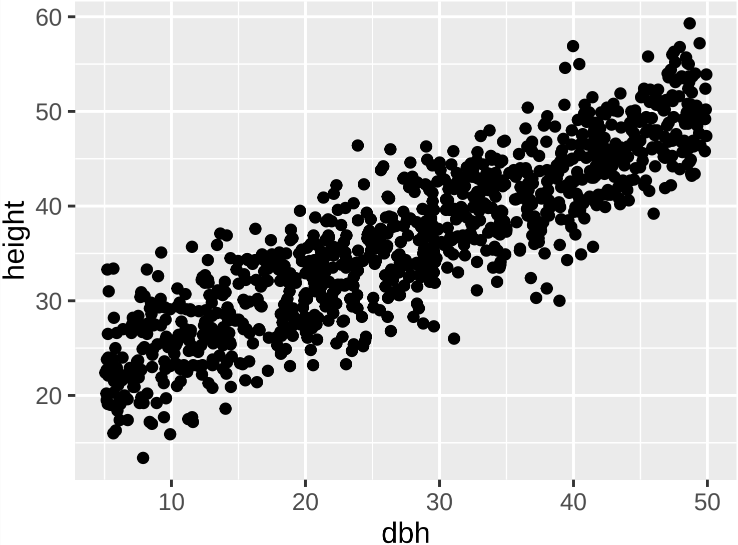
What variables as axes?

```
ggplot(trees, aes(x = dbh, y = height))
```



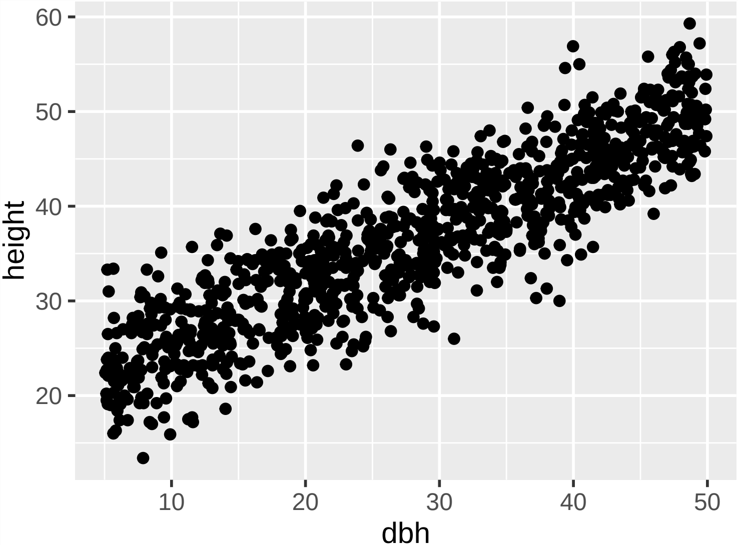
Adding layers (geoms)

```
ggplot(trees, aes(x = dbh, y = height)) +
  geom_point()
```



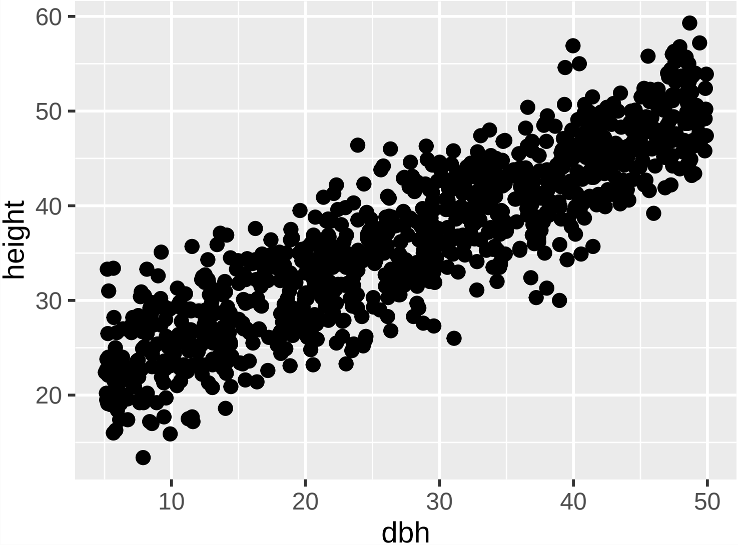
Adding layers (geoms)

```
myplot <- ggplot(trees, aes(x = dbh, y = height))
myplot + geom_point()
```



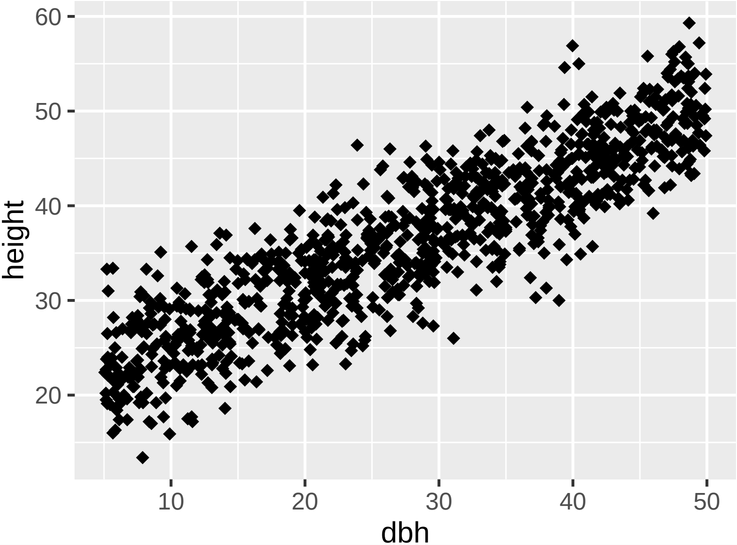
Changing point size and type

```
myplot + geom_point(size = 2)
```



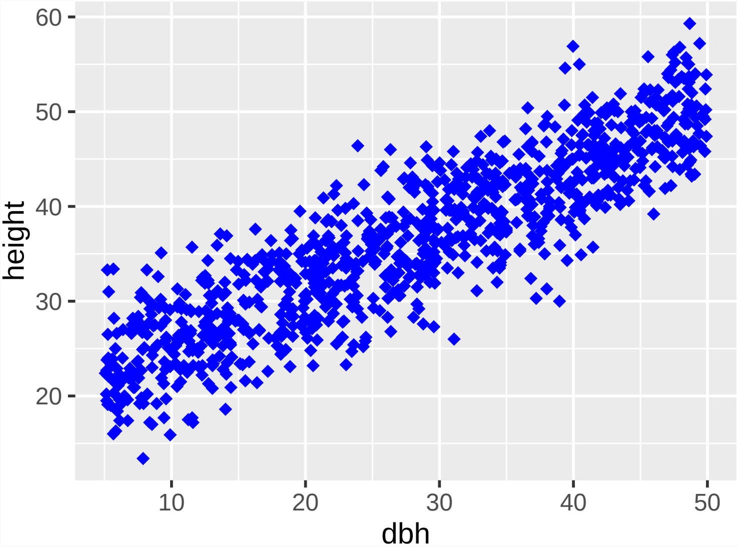
Changing point size and type

```
myplot + geom_point(size = 2, shape = 18)
```



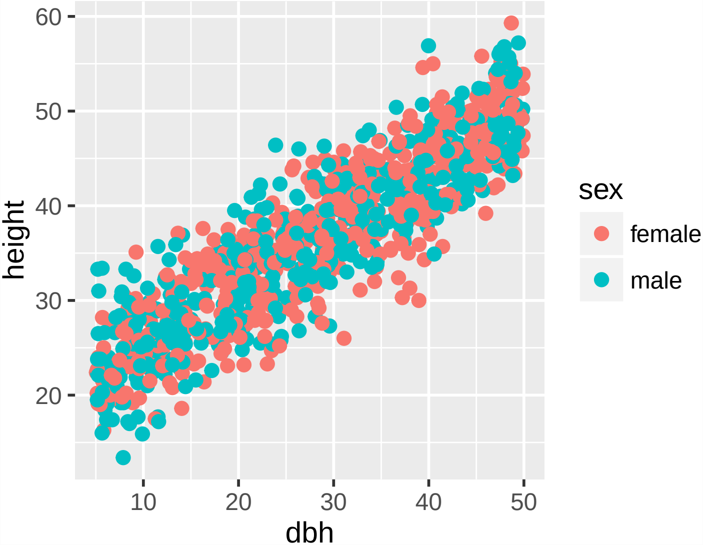
Changing point size and type

```
myplot + geom_point(size = 2, shape = 18, colour = "blue")
```



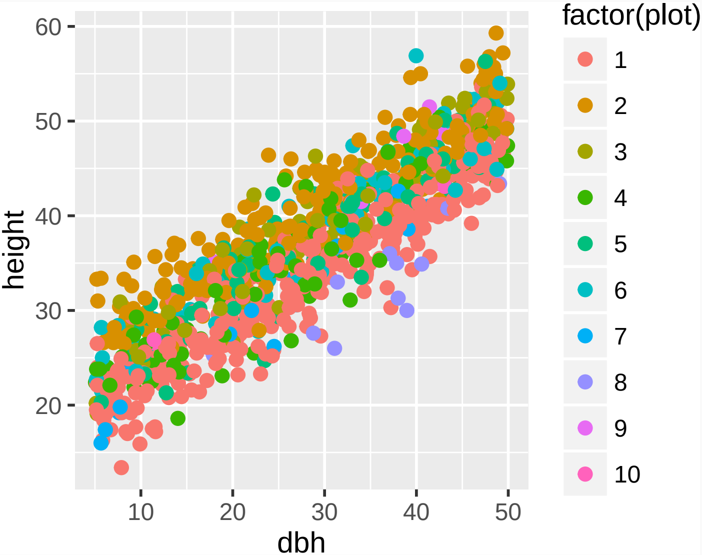
Map geom aesthetics to variable

```
myplot + geom_point(aes(colour = sex), size = 2)
```



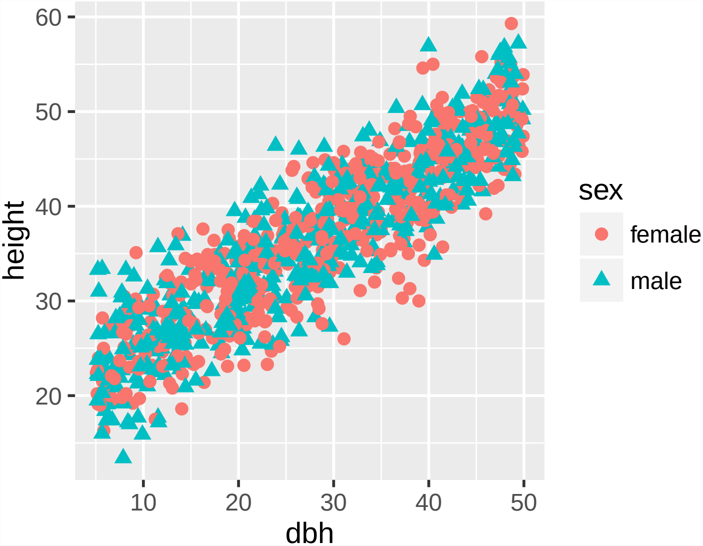
Map geom aesthetics to variable

```
myplot + geom_point(aes(colour = factor(plot)), size = 2)
```



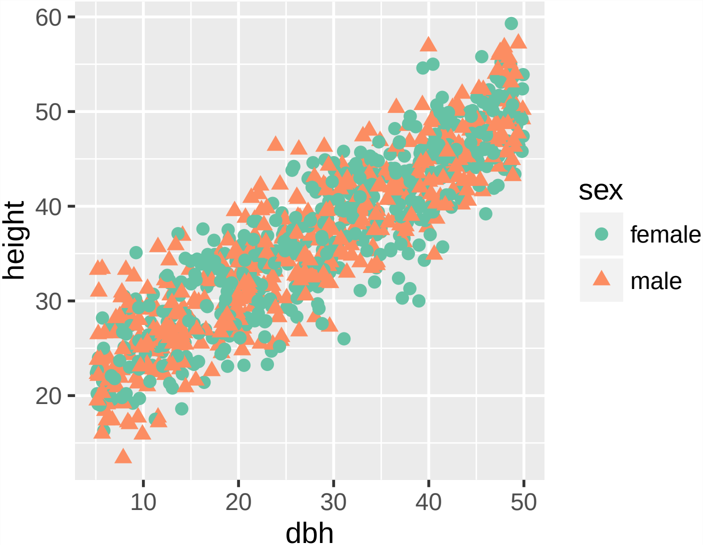
Map geom aesthetics to variable

```
myplot + geom_point(aes(colour = sex, shape = sex), size = 2)
```



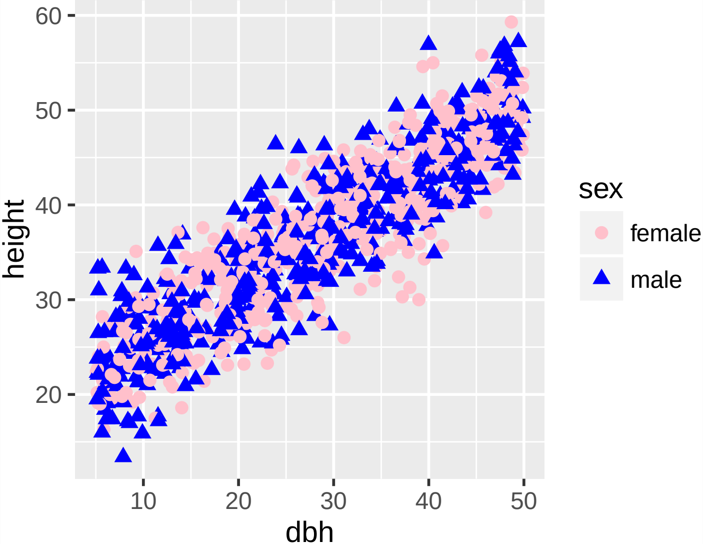
Change colour scale

```
myplot + geom_point(aes(colour = sex, shape = sex), size = 2) +
  scale_colour_brewer(type = "qual", palette = 7)
```



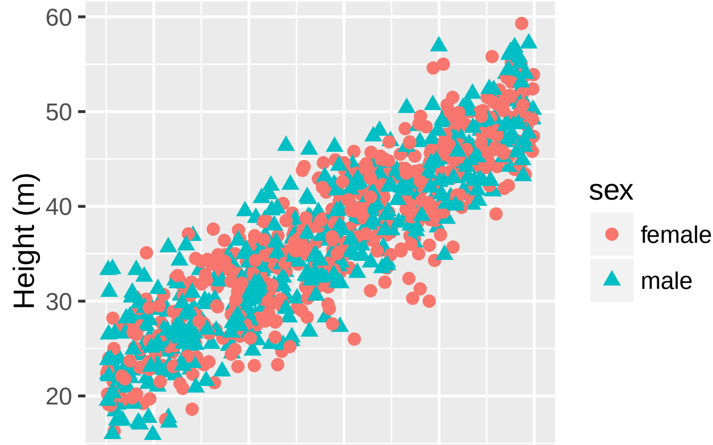
Change colour scale

```
myplot + geom_point(aes(colour = sex, shape = sex), size = 2) +
  scale_colour_manual(values = c("pink", "blue"))
```



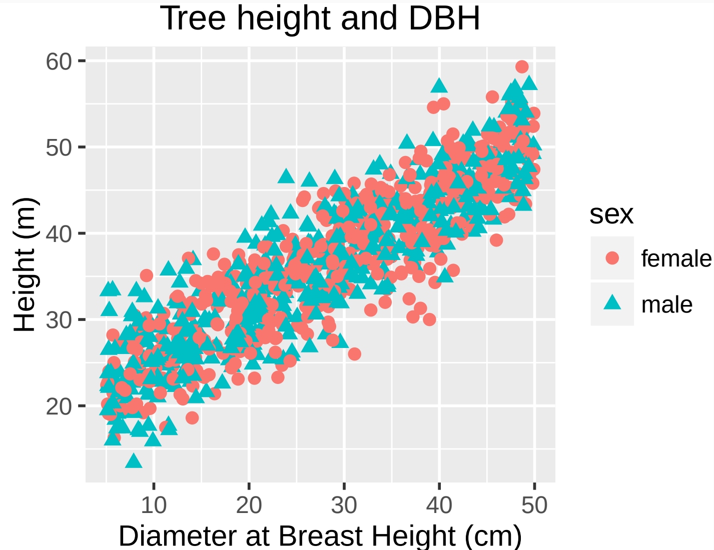
Change axis labels: xlab & ylab

```
myplot <- myplot + geom_point(aes(colour = sex, shape = sex), size = 2)
myplot <- myplot +
  xlab("Diameter at Breast Height (cm)") +
  ylab("Height (m)")
myplot
```



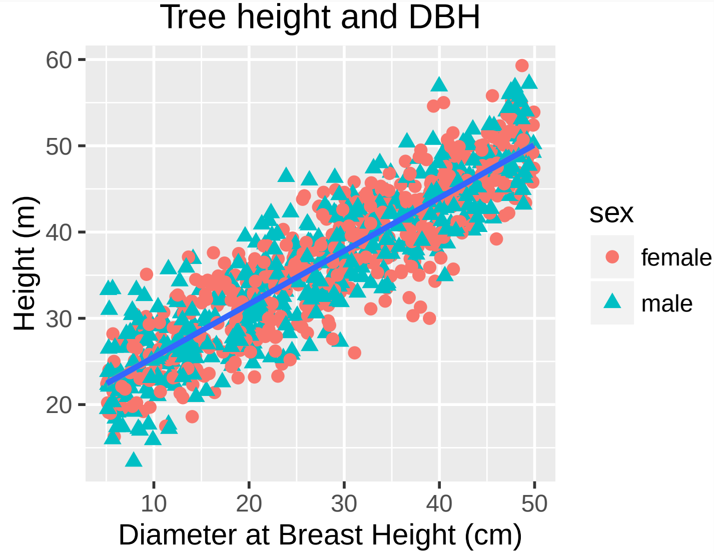
Set title: ggtitle

```
myplot <- myplot +
  ggtitle("Tree height and DBH")
```



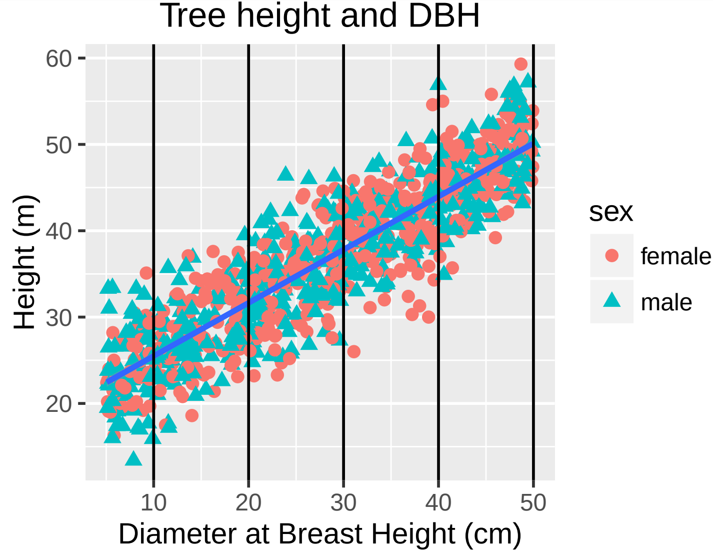
Adding another layer

```
myplot <- myplot + geom_smooth(method = "lm")
```

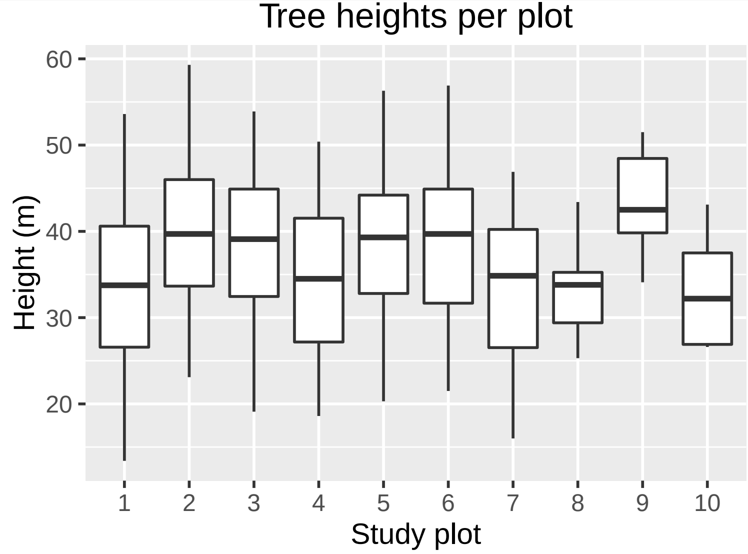


Adding another layer

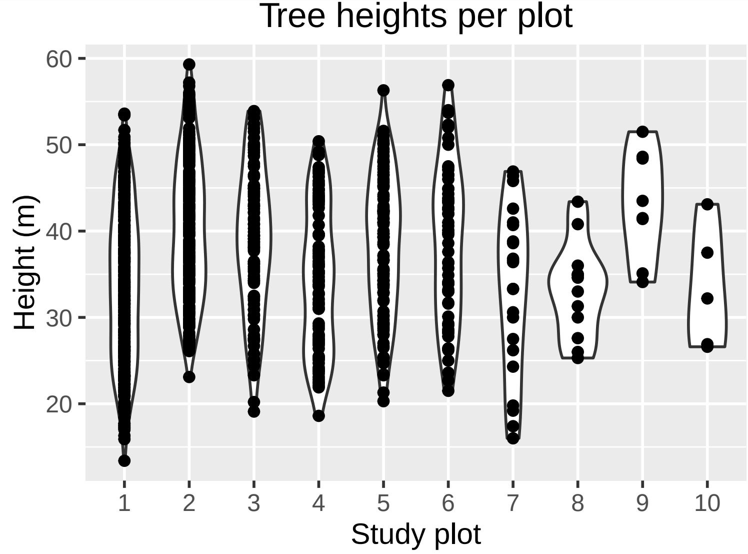
```
myplot + geom_vline(xintercept = c(10, 20, 30, 40, 50))
```



Exercise: Make a plot like this one



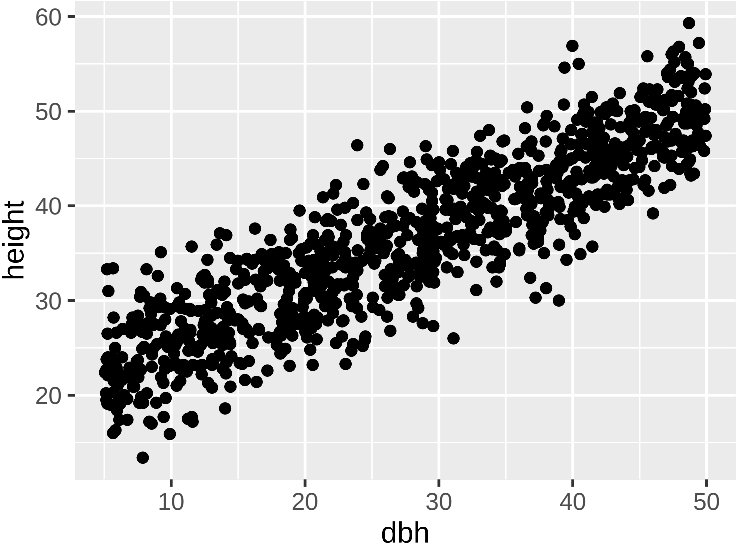
Exercise: Make a plot like this one



Themes: changing plot appearance

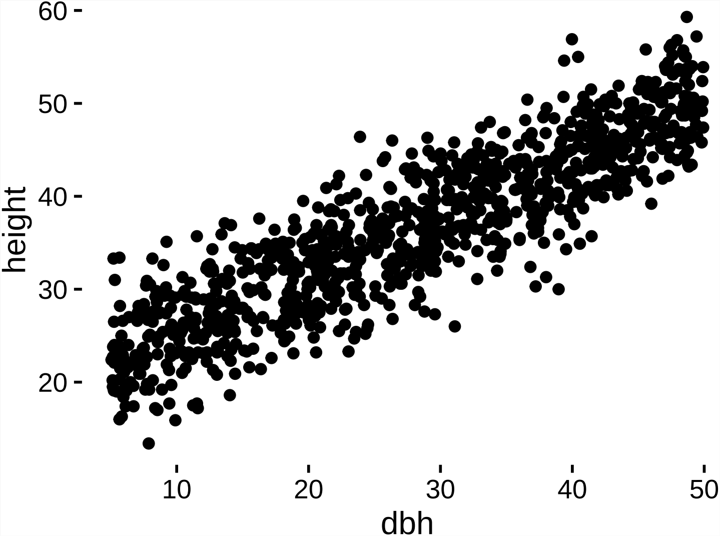
myplot

```
myplot <- ggplot(trees, aes(x = dbh, y = height)) +
  geom_point()
```



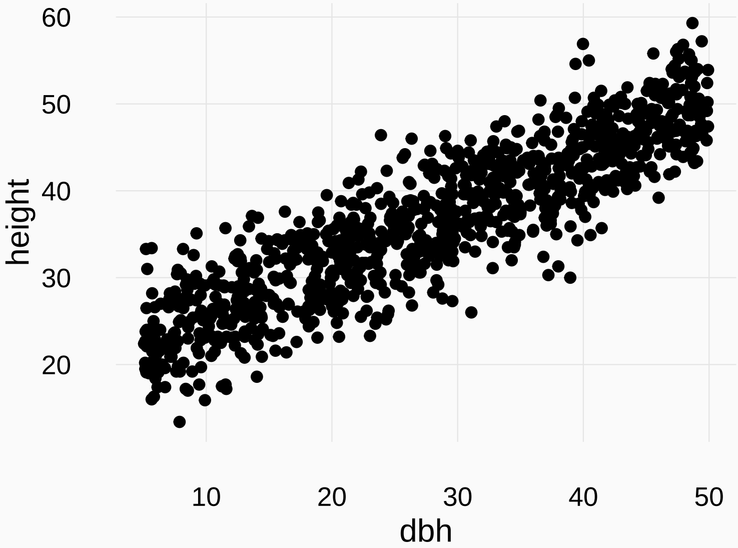
theme_classic

```
myplot + theme_classic()
```



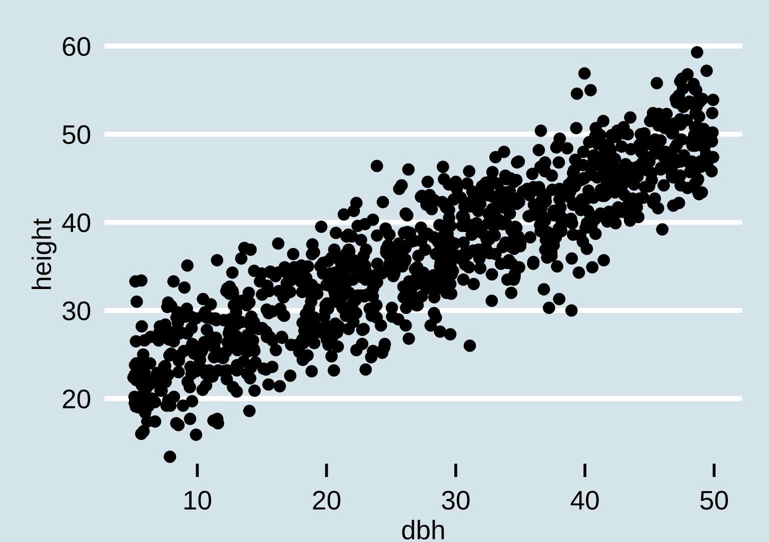
theme_minimal

```
myplot + theme_minimal()
```



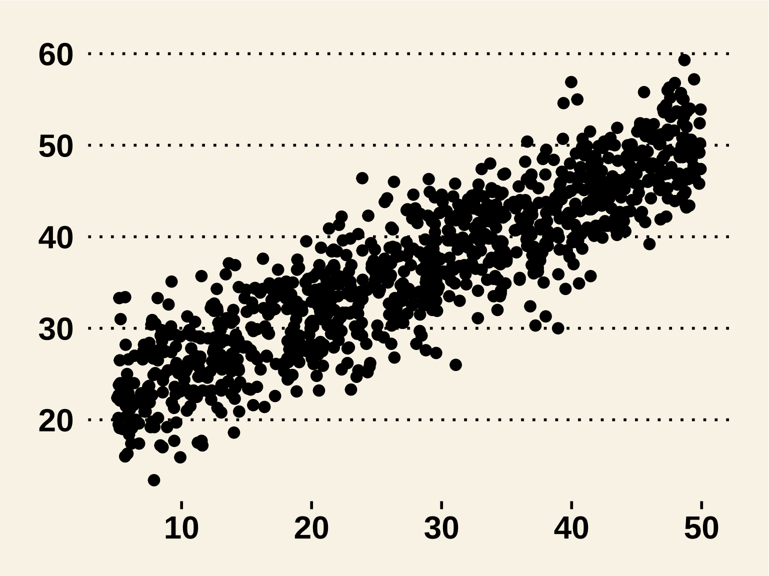
Lots of themes out there

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



Lots of themes out there

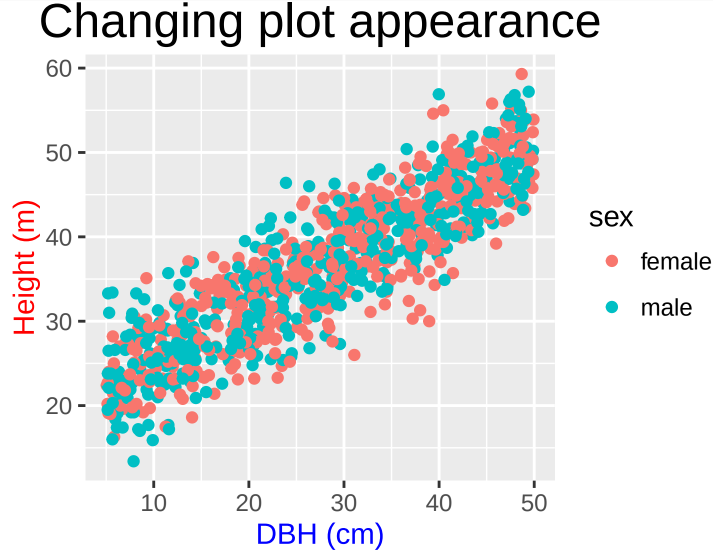
```
myplot + theme_wsj()
```



Editing themes

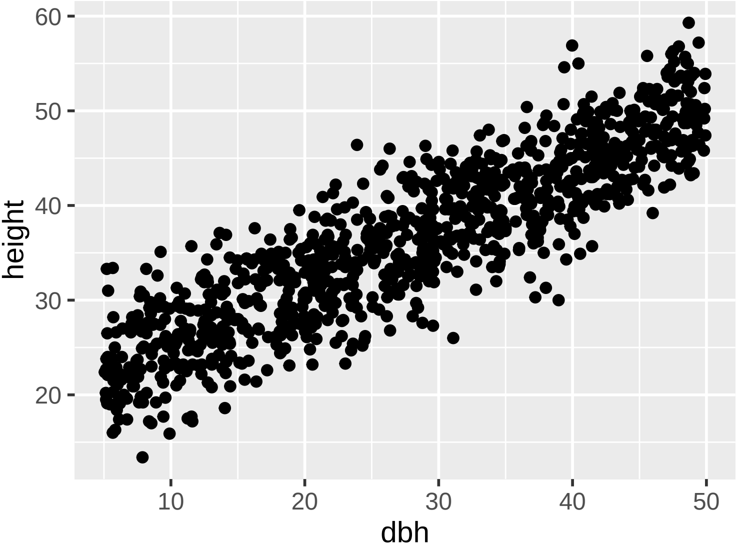
```
?theme
```

Exercise: make a plot like this one

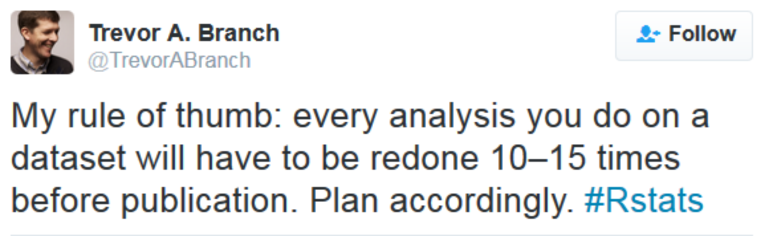


ggplot theme assistant (Rstudio addin)

https://github.com/calligross/ggthemeassist



Think twice before editing plots out of R



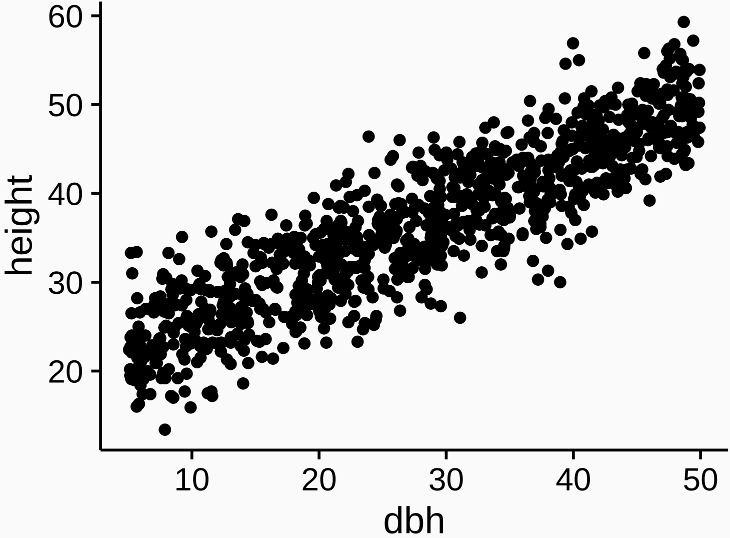
Trevor A. Branch
@TrevorABranch

My rule of thumb: every analysis you do on a dataset will have to be redone 10–15 times before publication. Plan accordingly. #Rstats

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

Publication-quality plots

```r
library(cowplot)
myplot
```

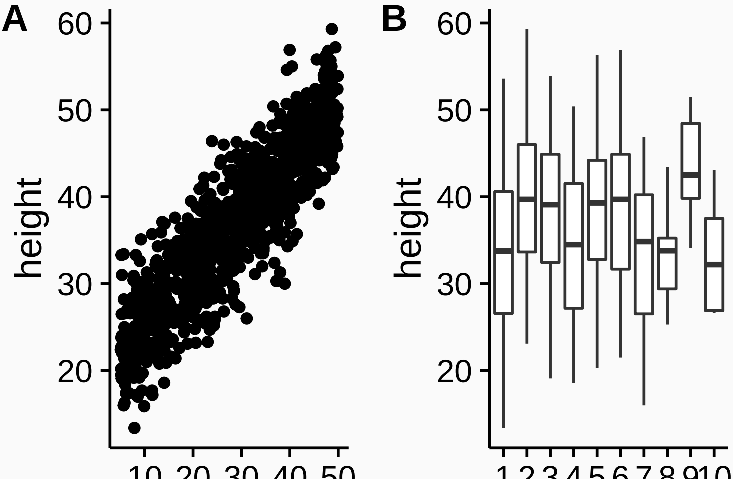


Publication themes:

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

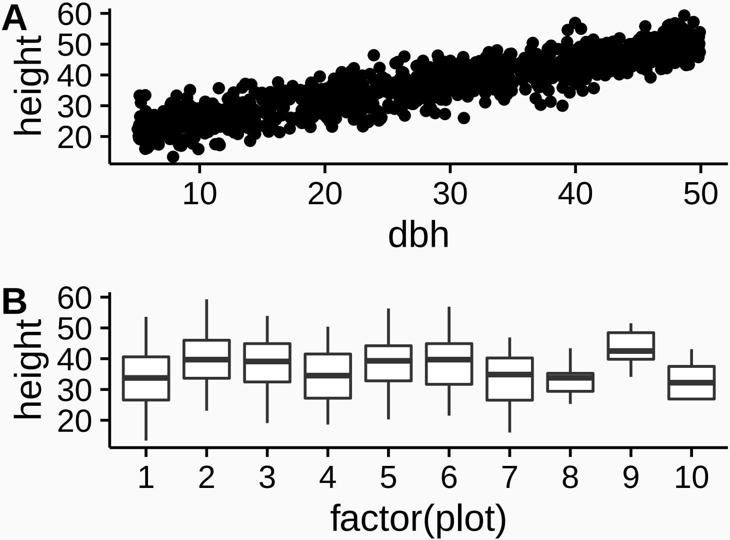
Composite figures

```
library(cowplot)
plot1 <- ggplot(trees, aes(dbh, height)) + geom_point()
plot2 <- ggplot(trees, aes(factor(plot), height)) + geom_boxplot()
plot_grid(plot1, plot2, labels = "AUTO")
```



Composite figures

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

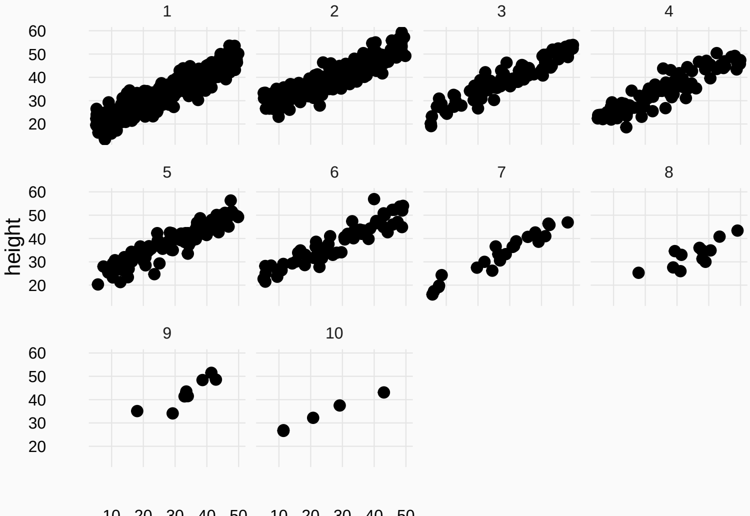


Saving plot: ggsave

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

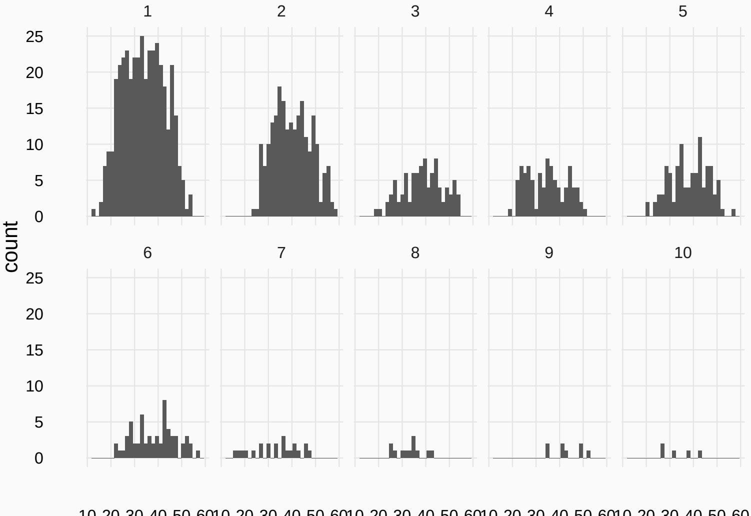
Facetting

```
ggplot(trees, aes(dbh, height)) +
  geom_point() + theme_minimal(base_size = 8) +
  facet_wrap(~plot)
```



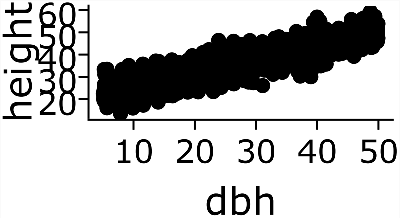
Facetting

```
ggplot(trees) +
  geom_histogram(aes(height)) + theme_minimal(base_size = 8) +
  facet_wrap(~plot, nrow = 2)
```



Interactivity

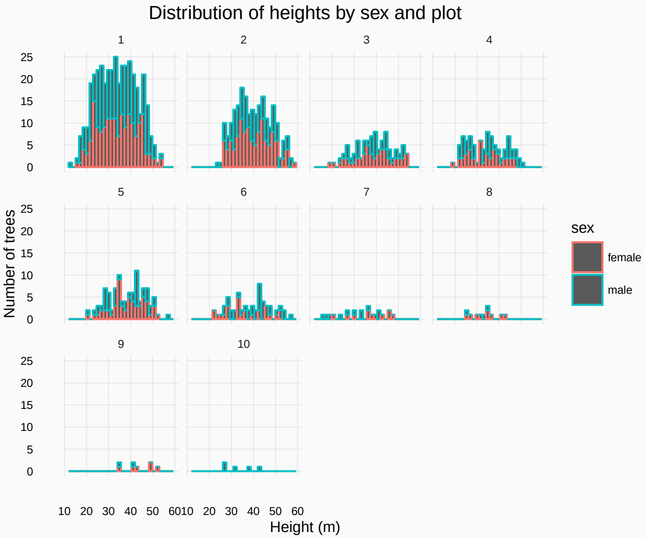
```
suppressPackageStartupMessages(library(plotly))
myplot <- ggplot(trees, aes(dbh, height)) + geom_point()
ggplotly(myplot)
```



Grammar of graphics

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

Exercise: make a plot like this one



END

Slides and source code available at

https://github.com/Pakillo/ggplot-intro