Statistical Analysis using R

IITA
Transforming African Agriculture

Graphics in R with ggplot2

Ibnou Dieng
Kayode Fowobaje
Sam Ofodile
Moshood Bakare
Oluwafemi Oyedele

IITA Biometrics Unit

01-02 & 07-09 December 2021

Course overview

1. ~~Short Introduction to R and RStudio~~
2. ~~Preparation of Data for Statistical Analysis~~
3. ~~Data wrangling~~
4. ~~Experimental Designs for Plant Breeding~~
5. ~~ANOVA and MET analysis~~
6. ~~Multivariate analysis~~
7. Graphics in R with ggplot2

Recap: Data Import

- To import a **csv** file into **R**

```
library(tidyverse)
mydata <- read_csv("mydata.csv")
```

- To import a **xlsx** file into **R**

```
library(readxl)
mydata <- read_excel("mydata.xlsx", sheet = "Sheet1")
```

Recap: Data Wrangling

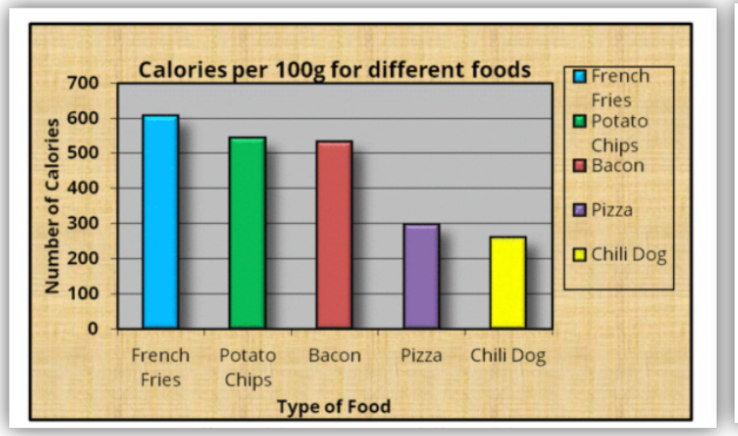
- There are five key **dplyr** functions in **tidyverse** to solve most of data manipulation challenges:
    - Pick observations by their values – **filter()**
    - Reorder (sort) the rows – **arrange()**
    - Pick variables by their names – **select()**
    - Create new variables with functions of existing variables – **mutate()**
    - Collapse many values down to a summary – **summarize()**

Introduction

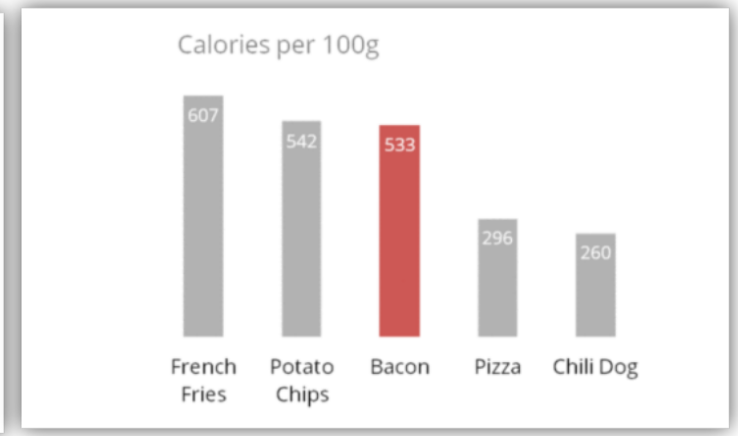
https://www.darkhorseanalytics.com/blog/data-looks-better-naked

Introduction

Before



After



Graphics: Less is more effective, attractive, impactive

Introduction

- R is a powerful programming language for graphics and visualizations
- Graphics in R can be done in three ways, via the:
  - `graphics` package: the base graphics in R, loaded by default
  - `lattice` package: which adds more functionalities to the base package
  - **`ggplot2`** package in the `tidyverse` package

The ggplot2

- The `graphics` package has large choice of plots: `plot`, `hist`, `barplot` , `boxplot`, `pie`, `mosaicplot`, etc. and additional features (e.g. `abline`, `lines`, `legend`, `mtext`, `rect`, etc.). Sometimes the preferred way to plotting for most R users
- `ggplot2` created by Hadley Wickham in 2005, is becoming one of the most popular **R** packages and the most popular package for graphics and data visualizations
- The `ggplot2` package is a much more modern approach to creating professional-quality graphics

The ggplot2

- The **gg**`plot2` package is based on the principles of the "**g**rammar of **g**raphics", a coherent system for describing and building graphs as a succession of layers
    - The **dataset** with the variables to represent. Done with the `ggplot()` function and comes first
    - The **variable(s)** to represent on the axes, and the aesthetic elements (such as `color`, `size`, `fill`, `shape` and `transparency`) of the objects to be represented. Done with the `aes()` function.
    - The **type of graphical representation** (`scatter plot`, `line plot`, `barplot`, `histogram`, etc.). Done with the functions `geom_point()`, `geom_line()`, `geom_bar()`, etc.
    - Additional layers (such as `labels`, `annotations`, `scales`, `axis ticks`, `legends`, `themes`, `facets`, etc.) can be added to **personalize** the plot

The ggplot2

- To create a plot, specify the data in the `ggplot()` function and add the required layers: `variables`, `aesthetic elements` and the `type of plot`:

```
ggplot(data) +
  aes(x = var_x, y = var_y) +
  geom_x()
```

- `data` in `ggplot()` is the name of the data frame which contains the variables `var_x` and `var_y`
- The `+` symbol used to indicate the different layers
- The layer `aes()` indicates the variables to be used in the plot and more generally, the aesthetic elements of the plot
- `x` in `geom_x()` represents the type of plot: `geom_point()`, `geom_line()`, etc.

The ggplot2: the data

- The `agridat` package with datasets from small-plot trials, multi-environment trials, uniformity trials, yield monitors, and more
- The `australia.soybean` is a multi-environment trial of 58 varieties of soybeans, in 4 locations across 2 years in Australia.
    - **env** (environment), 8 levels, location-year
    - **loc** (location)
    - **year**
    - **gen** (genotype) of soybeans,
    - **yield**, metric tons/hectare
    - **height**, meters
    - **lodging**
    - **size seed**, millimetres
    - **protein**, percentage
    - **oil**, percentage

The ggplot2: the data

- Data can be accessed using these commands

```
library(agridat)
data(australia.soybean)
dat <- australia.soybean
str(dat)
```

```
## 'data.frame':    464 obs. of  10 variables:
##  $ env    : Factor w/ 8 levels "B70","B71","L70",..: 3 3 3 3 3 3 3 3 3 3 .
##  $ loc    : Factor w/ 4 levels "Brookstead","Lawes",..: 2 2 2 2 2 2 2 2 2
##  $ year   : int  1970 1970 1970 1970 1970 1970 1970 1970 1970 1970 ...
##  $ gen    : Factor w/ 58 levels "G01","G02","G03",..: 1 2 3 4 5 6 7 8 9 10
##  $ yield  : num  2.39 2.28 2.57 2.88 2.39 ...
##  $ height : num  1.45 1.45 1.46 1.26 1.33 ...
##  $ lodging: num  4.25 4.25 3.75 3.5 3.5 4 3 3.25 3 3.75 ...
##  $ size   : num  8.45 9.95 10.85 10.05 11 ...
##  $ protein: num  36.7 37.5 37.8 38.5 37.5 ...
##  $ oil    : num  20.9 20.7 21.3 22 22.1 ...
```

The ggplot2: the data

- The data is available in the `agridat` package as a `data frame`. We have opted to work with `tibble`, let's convert this to a `tibble`

```
library(tidyverse)
dat <- as_tibble(dat)
dat
```

```
## # A tibble: 464 x 10
##    env   loc    year gen   yield height lodging  size protein   oil
##    <fct> <fct> <int> <fct> <dbl>  <dbl>   <dbl> <dbl>   <dbl> <dbl>
##  1 L70   Lawes  1970 G01    2.39  1.44     4.25  8.45    36.7  20.9
##  2 L70   Lawes  1970 G02    2.28  1.45     4.25  9.95    37.6  20.7
##  3 L70   Lawes  1970 G03    2.57  1.46     3.75 10.8     37.8  21.3
##  4 L70   Lawes  1970 G04    2.88  1.26     3.5  10.0     38.4  22.0
##  5 L70   Lawes  1970 G05    2.39  1.34     3.5  11       37.5  22.1
##  6 L70   Lawes  1970 G06    2.41  1.36     4    11.8     38.2  21.2
##  7 L70   Lawes  1970 G07    2.70  1.3      3    11.8     37.4  21.7
##  8 L70   Lawes  1970 G08    2.46  0.955    3.25 10       35.2  21.1
##  9 L70   Lawes  1970 G09    2.57  1.03     3    11.2     35.9  21.5
## 10 L70   Lawes  1970 G10    2.98  1.16     3.75 10.8     39.7  20.4
## # … with 454 more rows
```

The ggplot2: the data

- The env, loc, year, and gen are the factors but as year was considered as integer, let's convert it to a factor

```
library(tidyverse)
dat$year <- as.factor(dat$year)
dat
```

```
## # A tibble: 464 x 10
##    env   loc   year  gen   yield height lodging  size protein   oil
##    <fct> <fct> <fct> <fct> <dbl>  <dbl>   <dbl> <dbl>   <dbl> <dbl>
##  1 L70   Lawes 1970  G01    2.39  1.44     4.25  8.45    36.7  20.9
##  2 L70   Lawes 1970  G02    2.28  1.45     4.25  9.95    37.6  20.7
##  3 L70   Lawes 1970  G03    2.57  1.46     3.75 10.8     37.8  21.3
##  4 L70   Lawes 1970  G04    2.88  1.26     3.5  10.0     38.4  22.0
##  5 L70   Lawes 1970  G05    2.39  1.34     3.5  11       37.5  22.1
##  6 L70   Lawes 1970  G06    2.41  1.36     4    11.8     38.2  21.2
##  7 L70   Lawes 1970  G07    2.70  1.3      3    11.8     37.4  21.7
##  8 L70   Lawes 1970  G08    2.46  0.955    3.25 10       35.2  21.1
##  9 L70   Lawes 1970  G09    2.57  1.03     3    11.2     35.9  21.5
## 10 L70   Lawes 1970  G10    2.98  1.16     3.75 10.8     39.7  20.4
## # … with 454 more rows
```

The ggplot2: scatter plot

- A scatter plot is used to visualize the relation between two quantitative variables Often used to visualize a potential correlation between the two variables

```
library(tidyverse)
ggplot(dat) +
     aes(x = oil, y = yield) +
     geom_point()
```

- We can sometimes see the `aes()` inside the `ggplot()` function

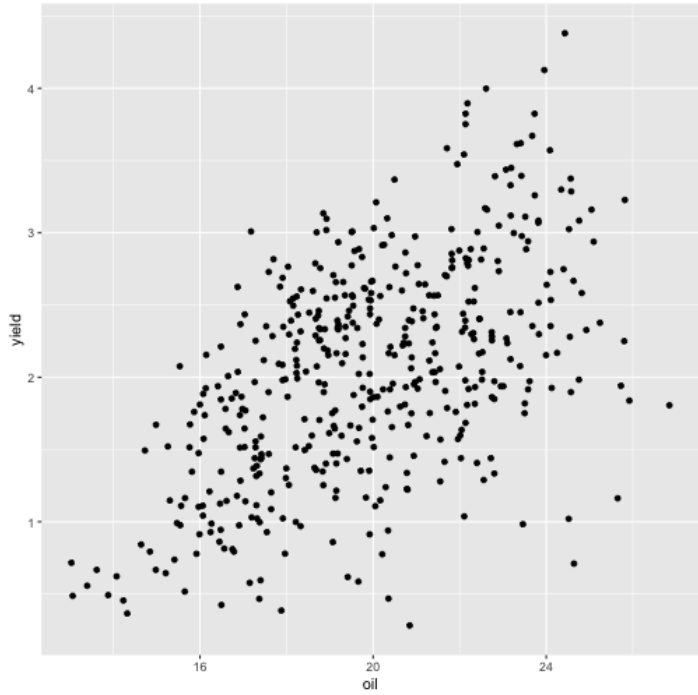
```
library(tidyverse)
ggplot(dat , aes(x = oil, y = yield)) +
  geom_point()
```

- I believe the first option (highlighted in yellow) is better for readability

The ggplot2: scatter plot

```
ggplot(dat) +
     aes(x = oil, y = yield) +
     geom_point()
```

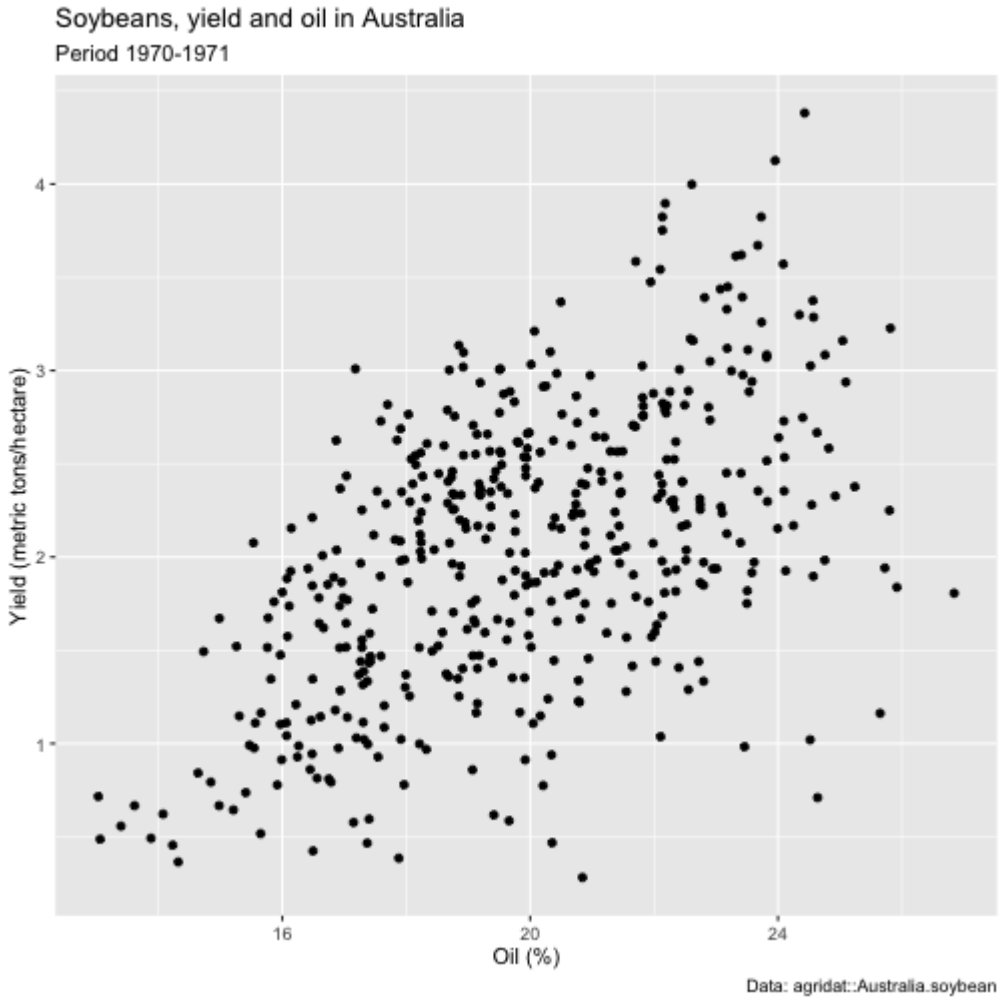
That's fine! But we can do better: personalize the plot to make it more informative. We can add a `title`, `subtitle`, `caption` and edit `axis labels` with the `labs()` function



The ggplot2: scatter plot

```
ggplot(dat) +
  aes(x = oil, y = yield) +
  geom_point() +
  labs(title = "Soybeans, yield and oil in Australia",
       subtitle = "Period 1970-1971",
       caption = "Data: agridat::Australia.soybean",
       x = "Oil (%)",
       y = "Yield (metric tons/hectare)"
       )
```

The ggplot2: scatter plot



The ggplot2: scatter plot

- We can save the "basic" plot in an object and add layer(s) by cascades for more readability

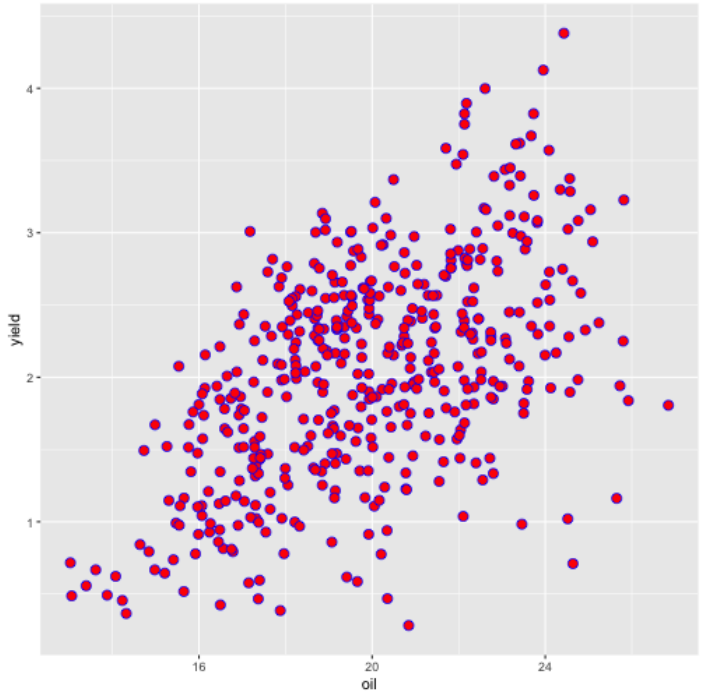
```
p <- ggplot(dat) +
  aes(x = oil, y = yield) +
  geom_point()

p <- p +
  labs(title = "Soybeans, yield and oil in Australia",
       subtitle = "Period 1970-1971",
       caption = "Data: agridat::Australia.soybean",
       x = "Oil (%)",
       y = "Yield (metric tons/hectare)"
       )
```

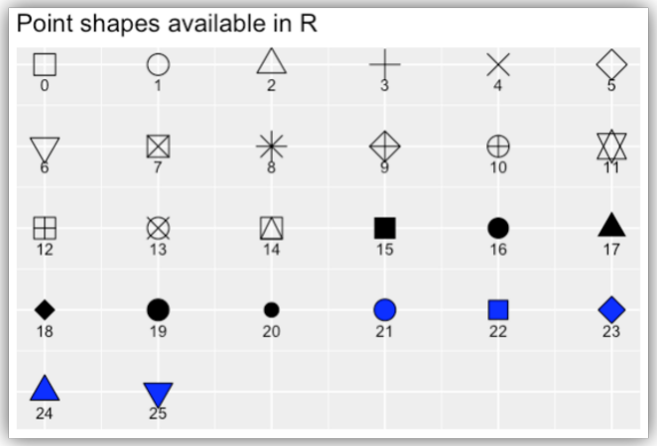
The ggplot2: scatter plot

- `geom_point()` understands also the following `aesthetics`
  - `alpha`
  - `colour`
  - `fill`
  - `shape`
  - `size`
  - `stroke`

The ggplot2: scatter plot

```
p <- ggplot(dat) +
  aes(x = oil, y = yield) +
  geom_point(shape=21, color="blue", fill="red", size=3)

p
```



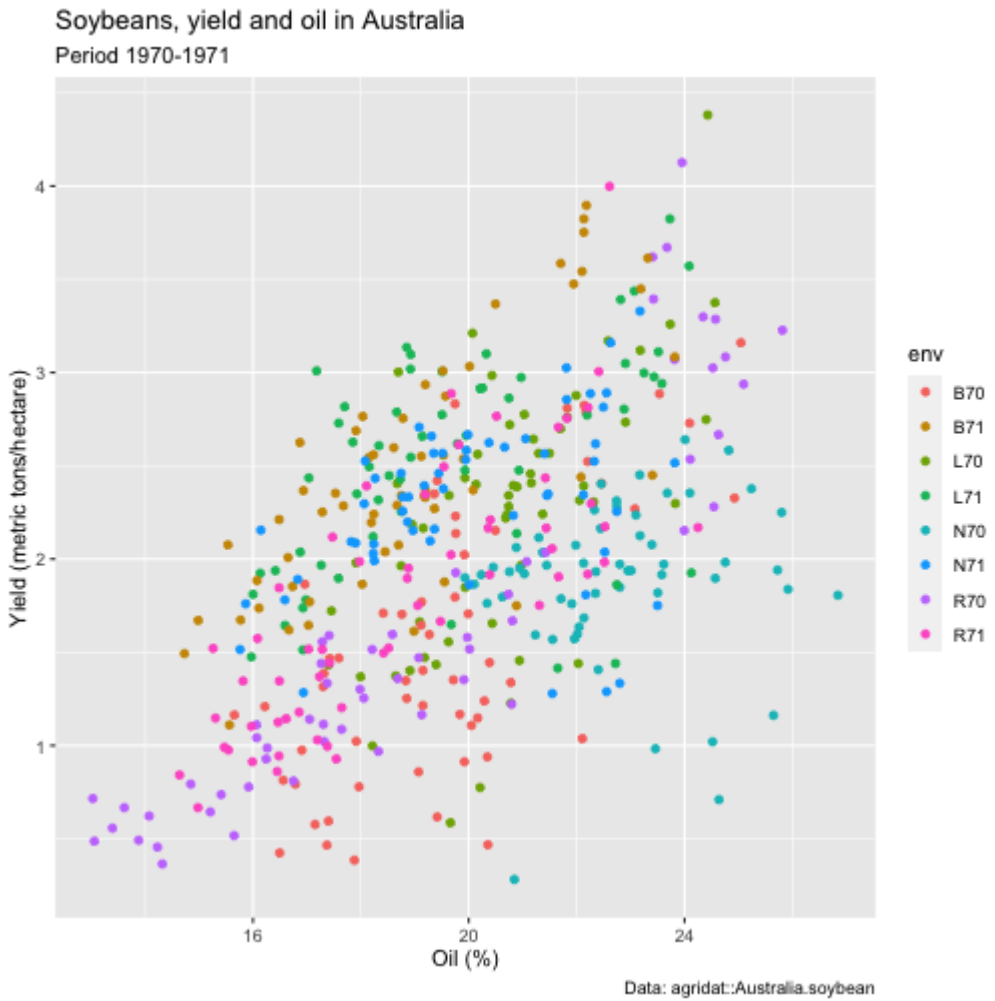


The ggplot2: scatter plot

The plot of the yield and oil content of the 58 varieties is across environments. We may be interested to see it by environment

```
p <- ggplot(dat) +
  aes(x = oil, y = yield, color=env) +
  geom_point()

p <- p +
  labs(title = "Soybeans, yield and oil in Australia",
       subtitle = "Period 1970-1971",
       caption = "Data: agridat::Australia.soybean",
       x = "Oil (%)",
       y = "Yield (metric tons/hectare)"
       )
p
```

The ggplot2: scatter plot

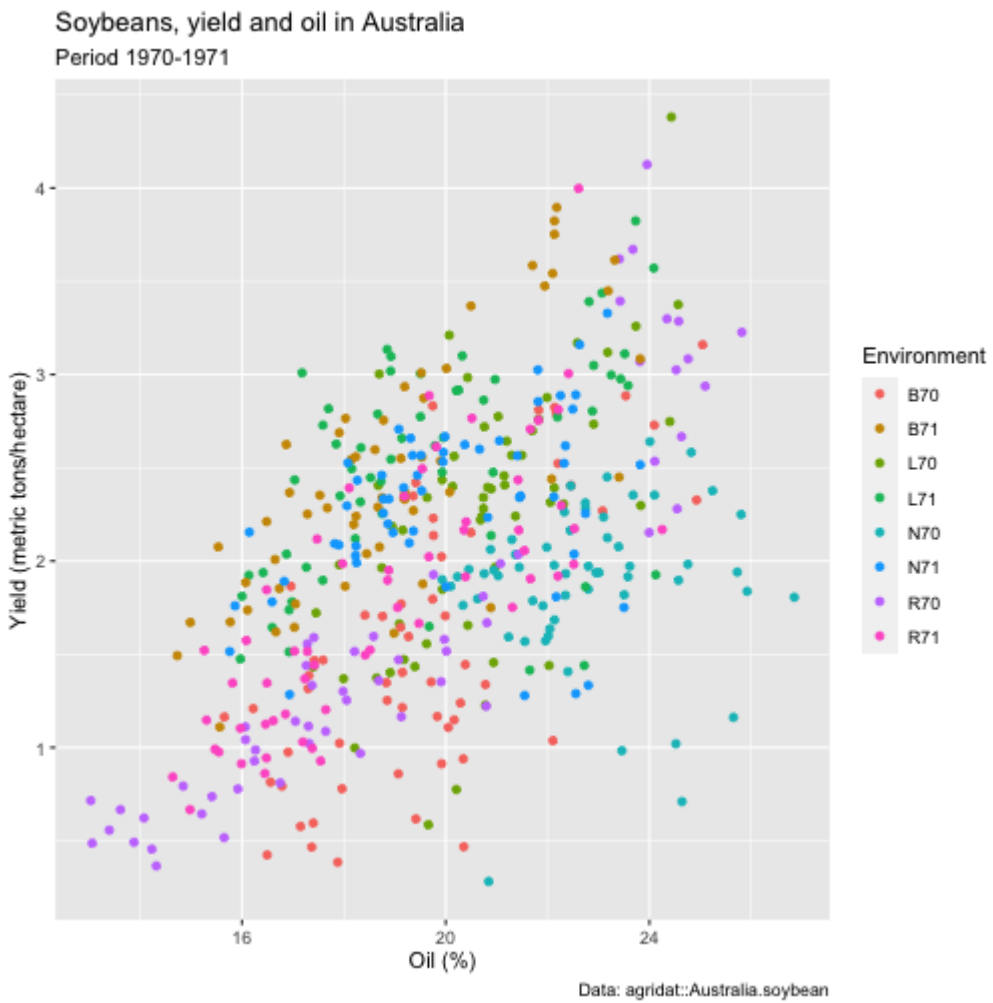


The ggplot2: scatter plot

We can do better by changing the title of the legend: “Environment” instead of “env”

```
p <- ggplot(dat) +
  aes(x = oil, y = yield, color=env) +
  geom_point()

p <- p +
  labs(title = "Soybeans, yield and oil in Australia",
       subtitle = "Period 1970-1971",
       caption = "Data: agridat::Australia.soybean",
       x = "Oil (%)",
       y = "Yield (metric tons/hectare)"
       )
p <- p + scale_color_discrete(name="Environment")
p
```

The ggplot2: scatter plot



The ggplot2: scatter plot

- We can change the default color
- The `RColorBrewer` package makes it easy to quickly load sensible color palettes

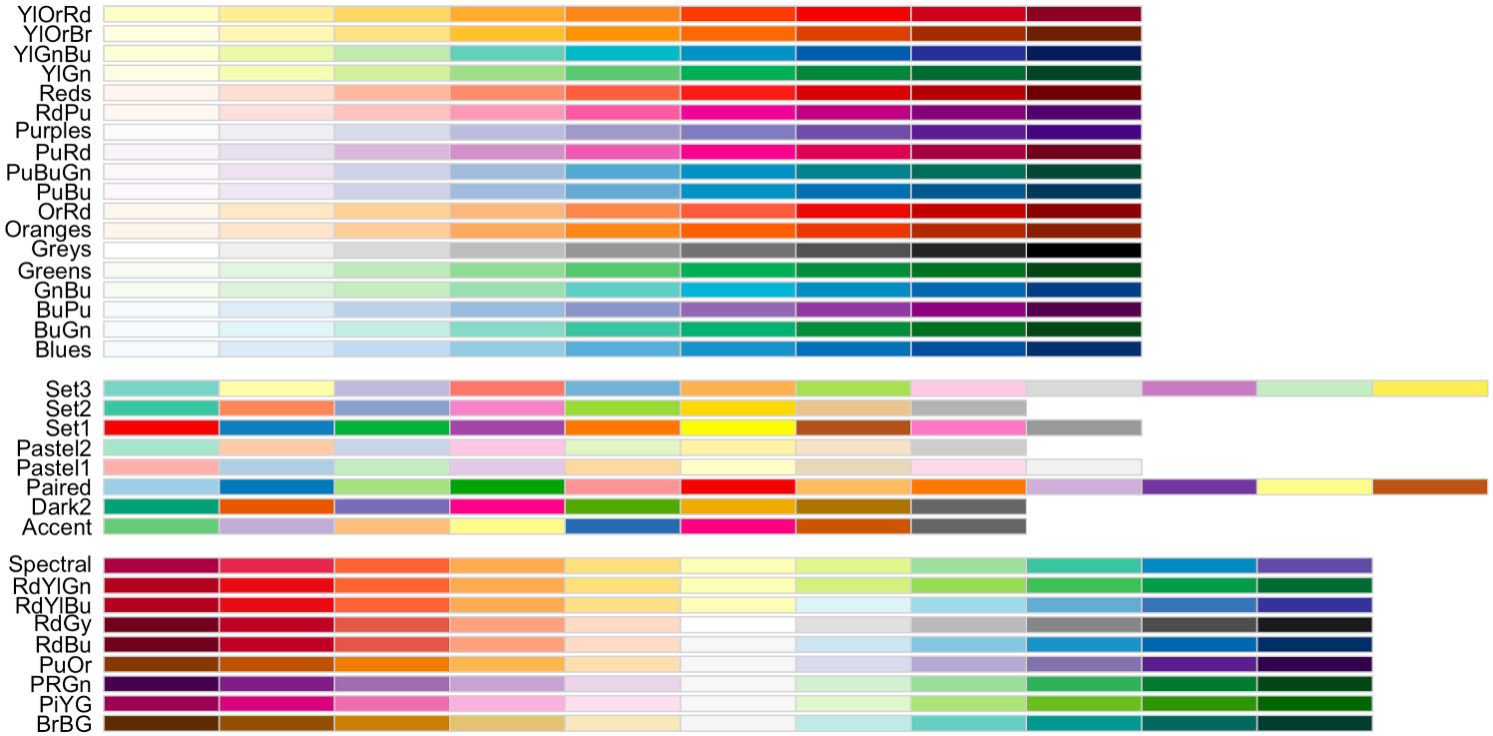
```
library(RColorBrewer)
```

- Three types of palettes: `sequential`, `diverging` and `qualitative`
    - Sequential: palettes are suited to ordered data that progress from low to high.
    - Diverging: palettes are suited to centered data with extremes in either direction.
    - Qualitative: palettes are suited to nominal or categorical data

The ggplot2: scatter plot

- The function below displays the available palettes

```
display.brewer.all()
```

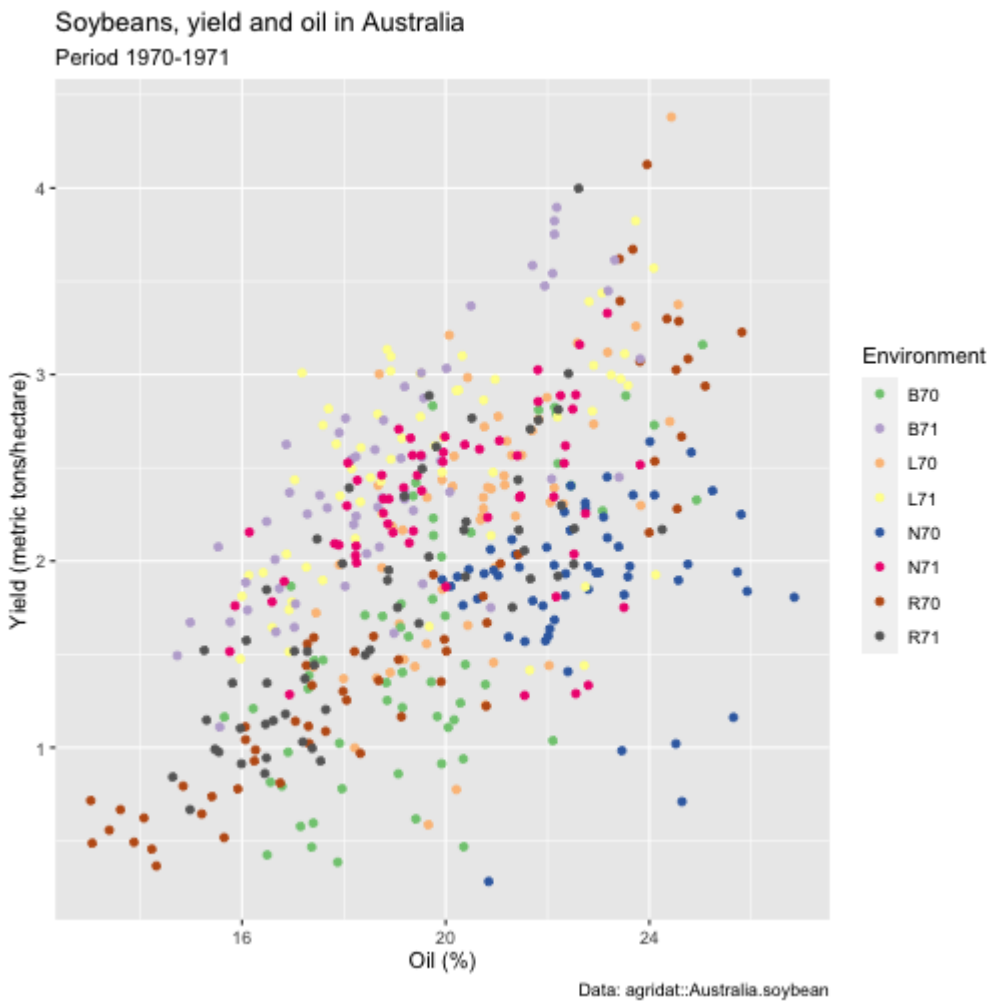


The ggplot2: scatter plot

We can change the palette via the function `scale_color_brewer()`

```
p <- ggplot(dat) +
  aes(x = oil, y = yield, color=env) +
  geom_point()

p <- p +
  labs(title = "Soybeans, yield and oil in Australia",
       subtitle = "Period 1970-1971",
       caption = "Data: agridat::Australia.soybean",
       x = "Oil (%)",
       y = "Yield (metric tons/hectare)"
       )
p <- p + scale_color_brewer(name="Environment", palette="Accent")
p
```

The ggplot2: scatter plot

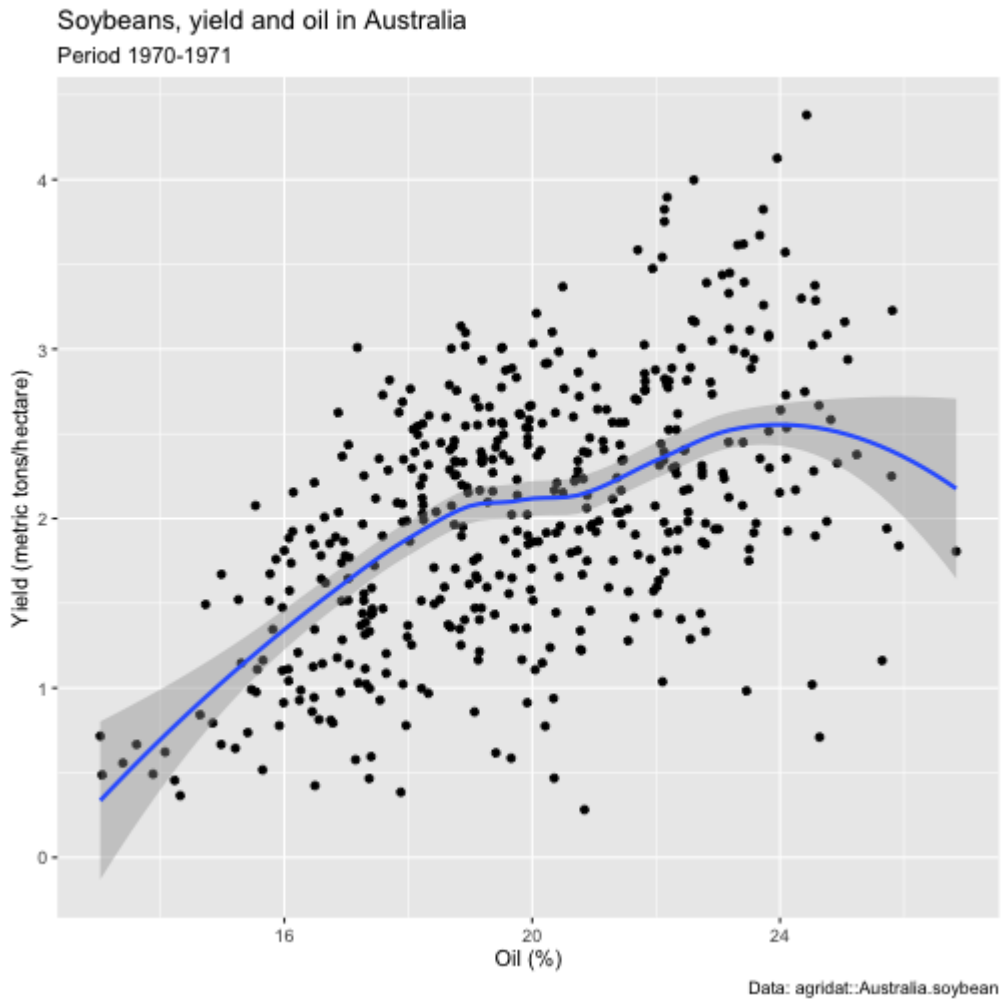


The ggplot2: scatter plot

Possible to add a `smooth` line fitted to the data

```
p <- ggplot(dat) +
  aes(x = oil, y = yield) +
  geom_point()

p <- p +
  labs(title = "Soybeans, yield and oil in Australia",
       subtitle = "Period 1970-1971",
       caption = "Data: agridat::Australia.soybean",
       x = "Oil (%)",
       y = "Yield (metric tons/hectare)"
       )
p <- p + geom_smooth()
p
```

The ggplot2: scatter plot

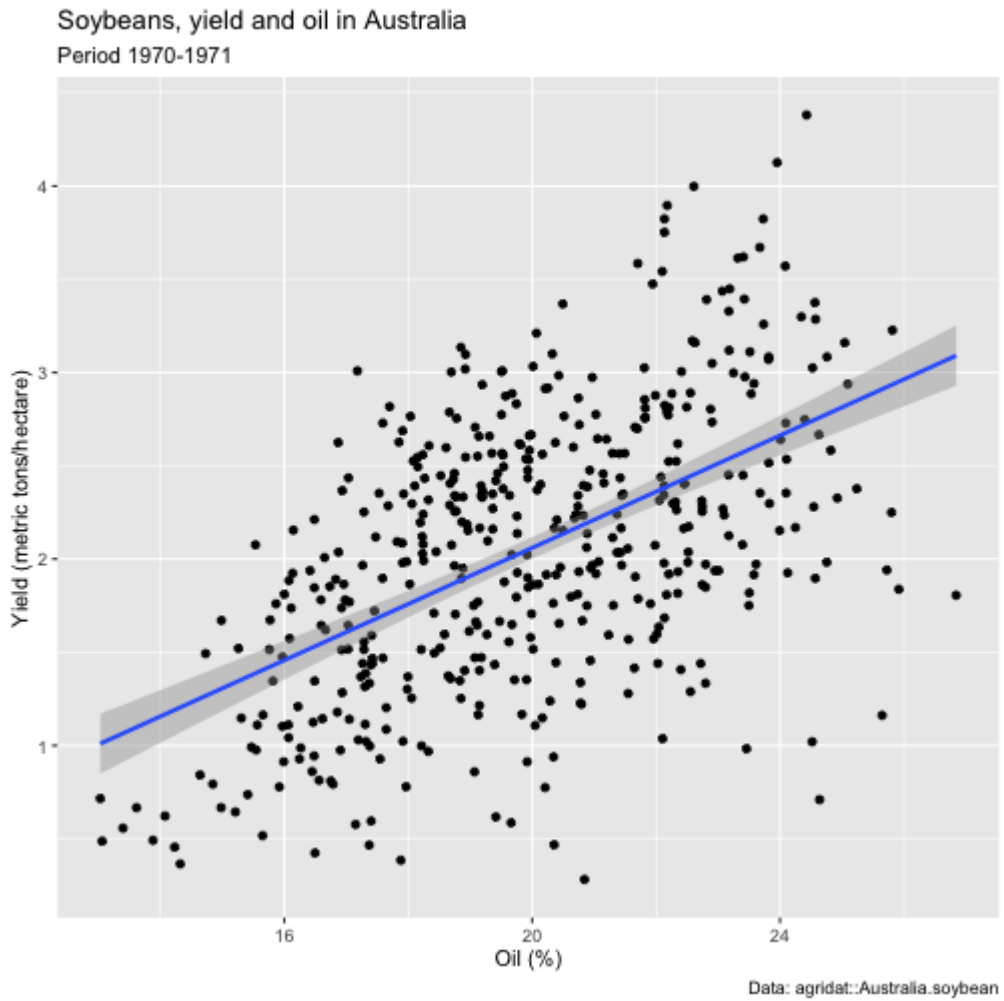


The ggplot2: scatter plot

In case of simple linear regression, possible to display the regression line on the plot. Can be done by adding `method = lm` in `geom_smooth()`

```
p <- ggplot(dat) +
  aes(x = oil, y = yield) +
  geom_point()

p <- p +
  labs(title = "Soybeans, yield and oil in Australia",
       subtitle = "Period 1970-1971",
       caption = "Data: agridat::Australia.soybean",
       x = "Oil (%)",
       y = "Yield (metric tons/hectare)"
       )
p <- p + geom_smooth(method = lm)
p
```

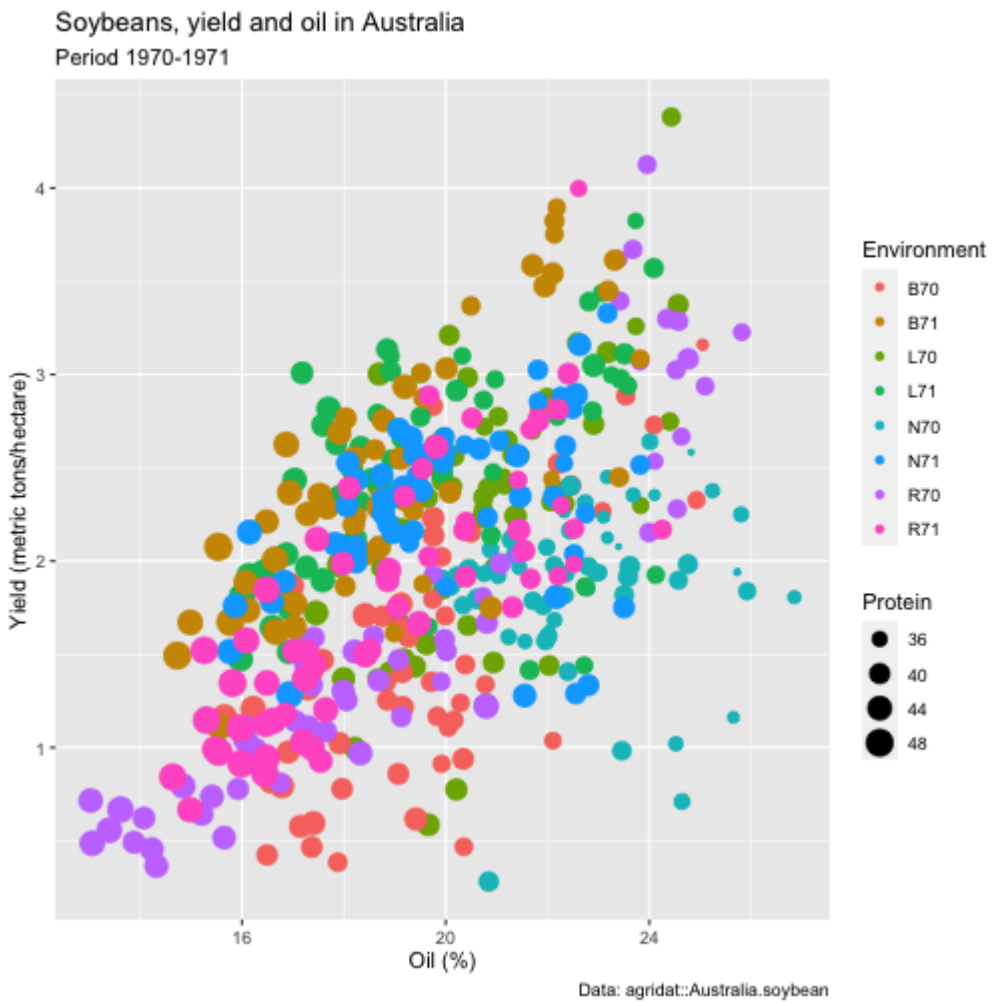
The ggplot2: scatter plot



The ggplot2: scatter plot

```
p <- ggplot(dat) +
  aes(x = oil, y = yield, color=env, size=protein) +
  geom_point()

p <- p +
  labs(title = "Soybeans, yield and oil in Australia",
       subtitle = "Period 1970-1971",
       caption = "Data: agridat::Australia.soybean",
       x = "Oil (%)",
       y = "Yield (metric tons/hectare)"
       )
p <- p + scale_color_discrete(name = "Environment")
p <- p + scale_size_continuous("Protein")
p
```

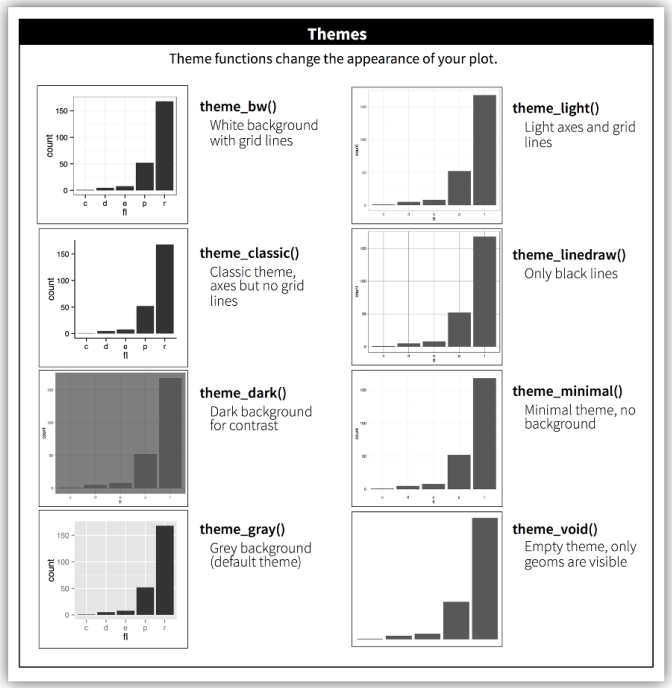
The ggplot2: scatter plot



The ggplot2: scatter plot

There are several functions in `ggplot2` to change the theme of the plot

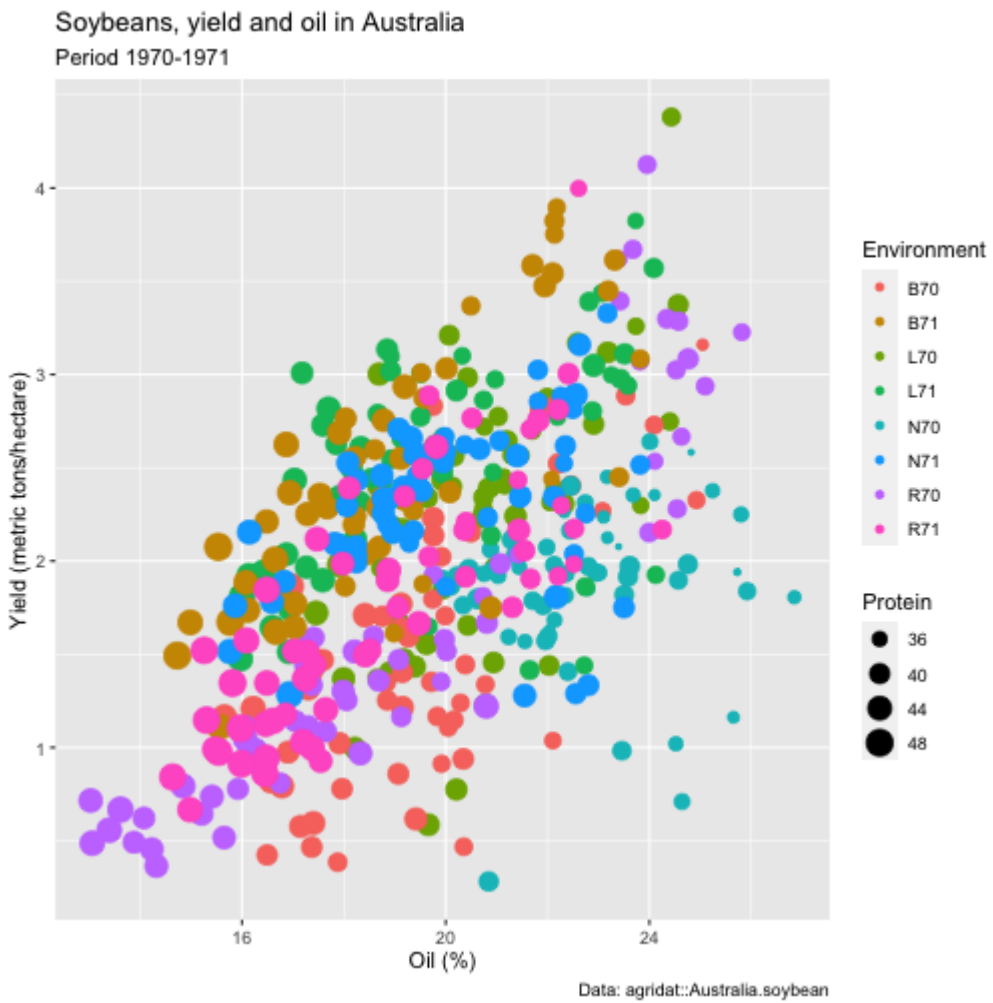
- `theme_gray()`: default
- `theme_bw()`: black and white
- `theme_minimal()`: minimal
- `theme_classic()`: classic



The ggplot2: scatter plot

```
p <- ggplot(dat) +
  aes(x = oil, y = yield, color=env, size=protein) +
  geom_point()

p <- p +
  labs(title = "Soybeans, yield and oil in Australia",
       subtitle = "Period 1970-1971",
       caption = "Data: agridat::Australia.soybean",
       x = "Oil (%)",
       y = "Yield (metric tons/hectare)"
       )
p <- p + scale_color_discrete(name = "Environment")
p <- p + scale_size_continuous("Protein")
p <- p + theme_gray()
p
```

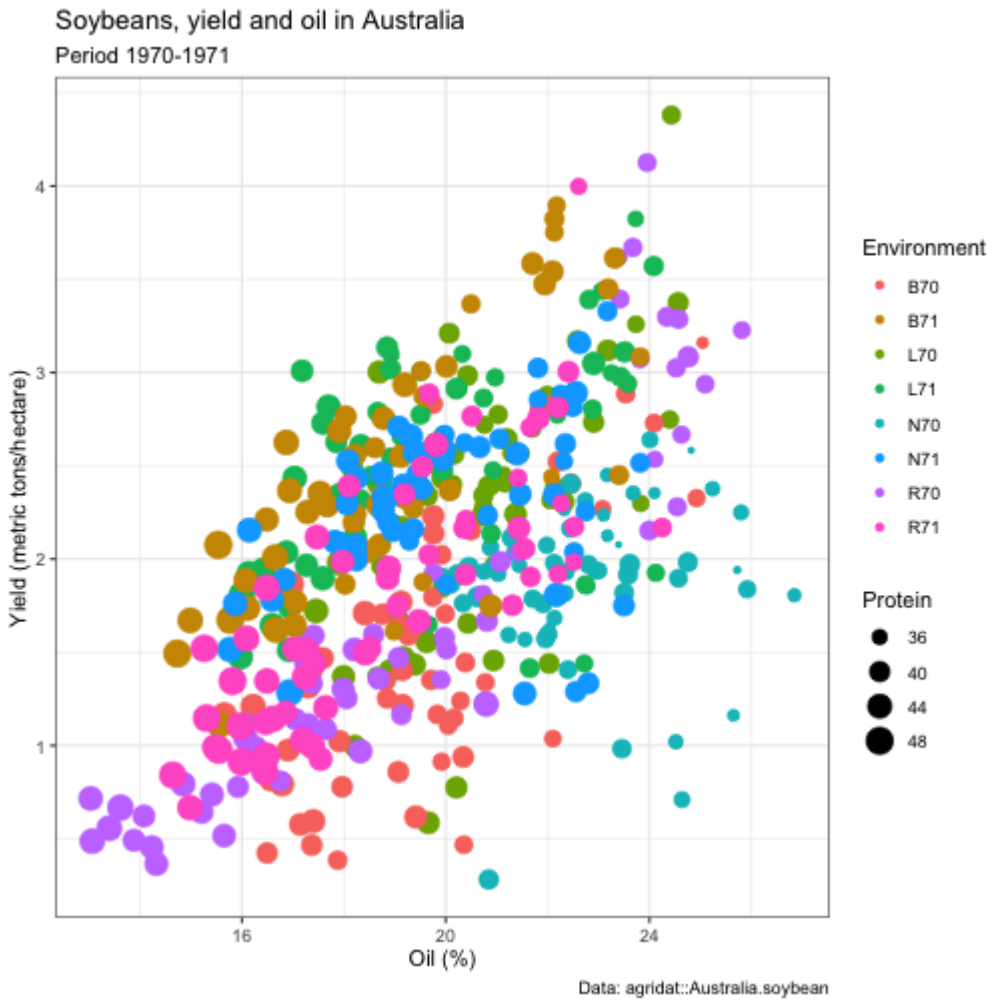
The ggplot2: scatter plot



The ggplot2: scatter plot

```
p <- ggplot(dat) +
  aes(x = oil, y = yield, color=env, size=protein) +
  geom_point()

p <- p +
  labs(title = "Soybeans, yield and oil in Australia",
       subtitle = "Period 1970-1971",
       caption = "Data: agridat::Australia.soybean",
       x = "Oil (%)",
       y = "Yield (metric tons/hectare)"
       )
p <- p + scale_color_discrete(name = "Environment")
p <- p + scale_size_continuous("Protein")
p <- p + theme_bw()
p
```

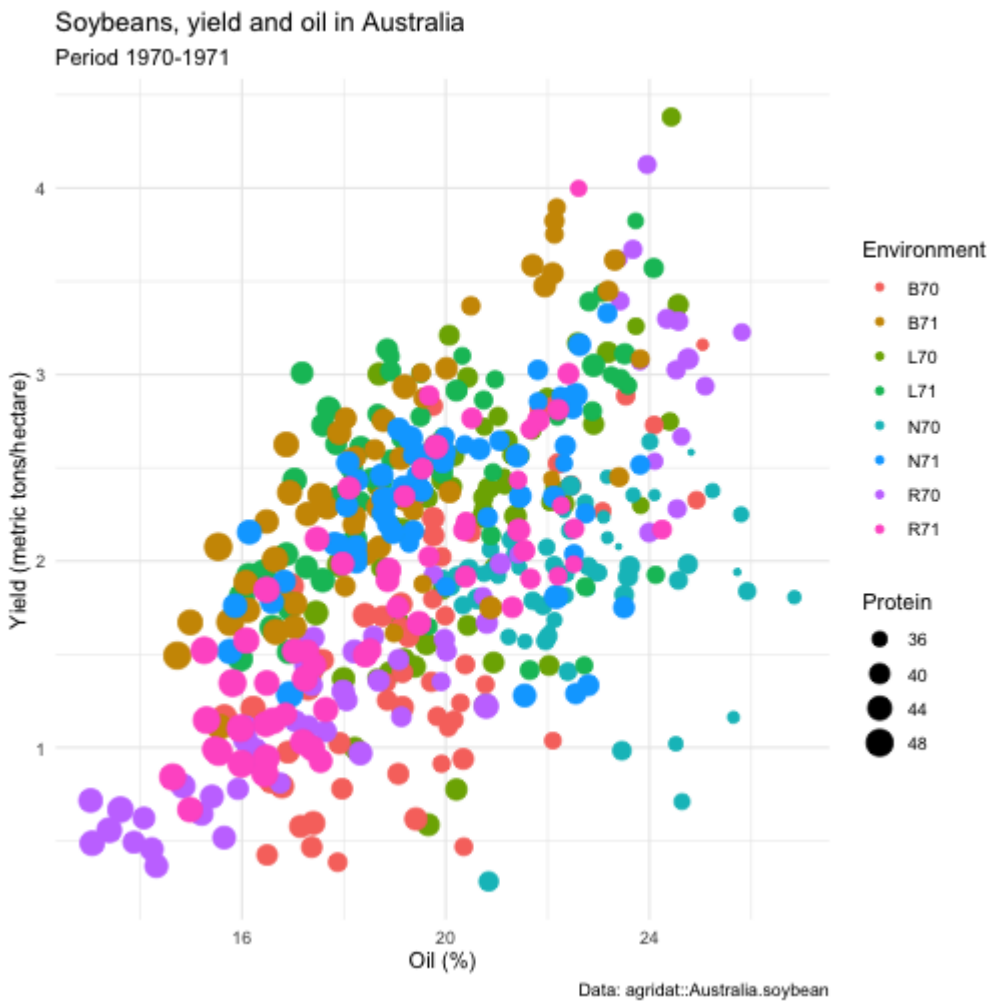
The ggplot2: scatter plot



The ggplot2: scatter plot

```
p <- ggplot(dat) +
  aes(x = oil, y = yield, color=env, size=protein) +
  geom_point()

p <- p +
  labs(title = "Soybeans, yield and oil in Australia",
       subtitle = "Period 1970-1971",
       caption = "Data: agridat::Australia.soybean",
       x = "Oil (%)",
       y = "Yield (metric tons/hectare)"
       )
p <- p + scale_color_discrete(name = "Environment")
p <- p + scale_size_continuous("Protein")
p <- p + theme_minimal()
p
```

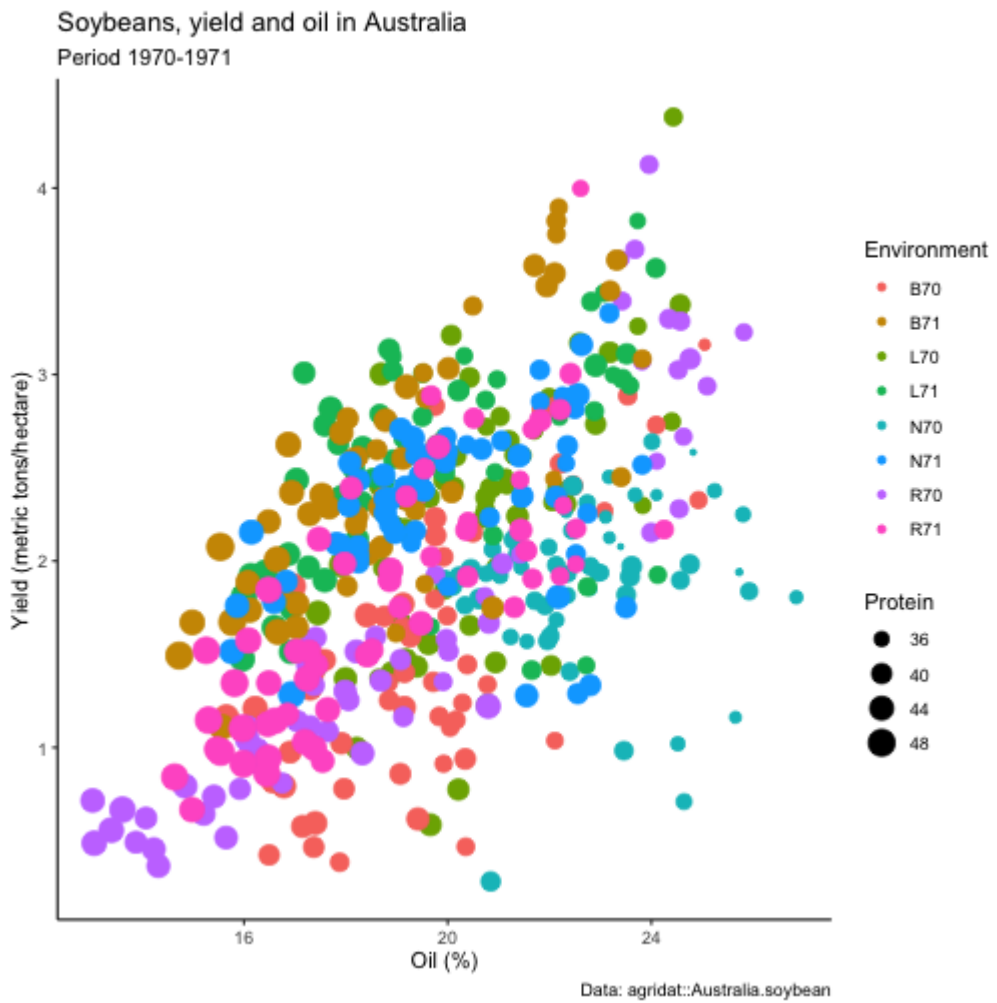
The ggplot2: scatter plot



The ggplot2: scatter plot

```
p <- ggplot(dat) +
  aes(x = oil, y = yield, color=env, size=protein) +
  geom_point()

p <- p +
  labs(title = "Soybeans, yield and oil in Australia",
       subtitle = "Period 1970-1971",
       caption = "Data: agridat::Australia.soybean",
       x = "Oil (%)",
       y = "Yield (metric tons/hectare)"
       )
p <- p + scale_color_discrete(name = "Environment")
p <- p + scale_size_continuous("Protein")
p <- p + theme_classic()
p
```

The ggplot2: scatter plot

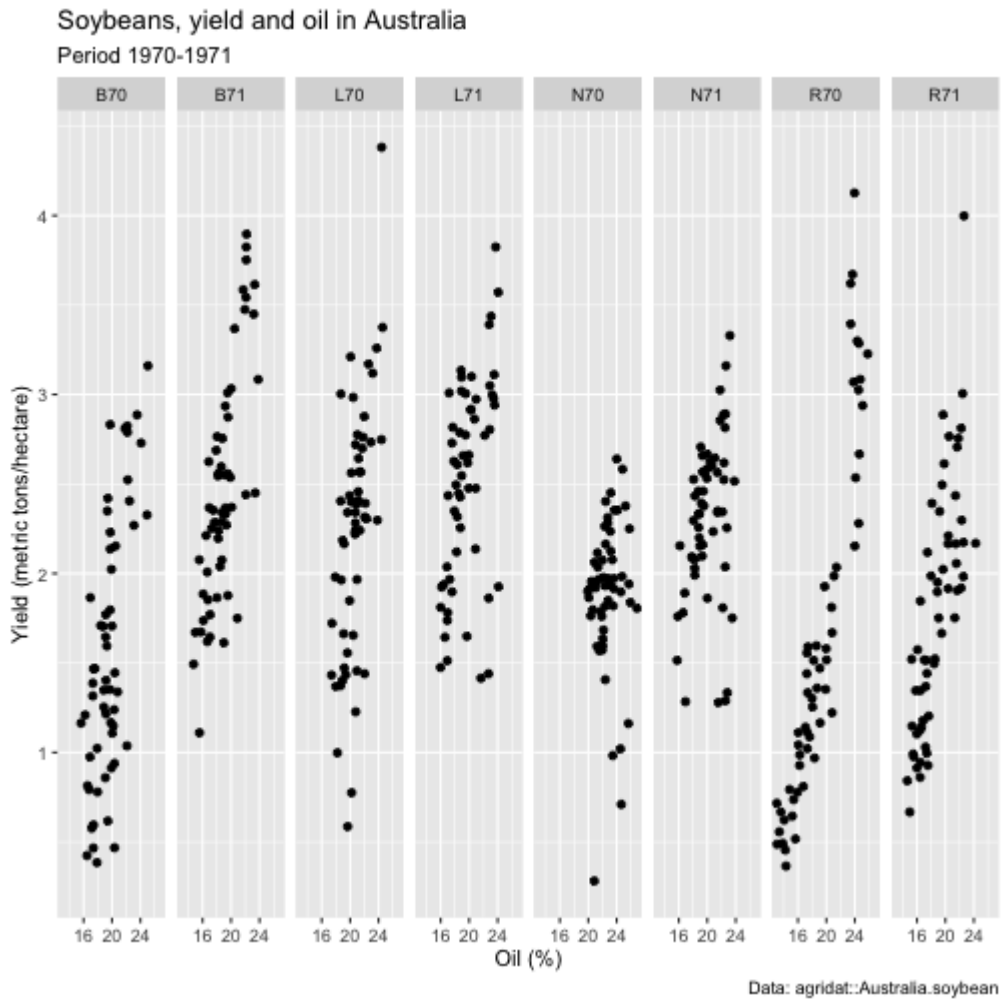


The ggplot2: scatter plot

Instead of plotting all the environments in the same plot and using color to differentiate them, we can use `facet_grid()` to divide the same graphic into several panels according to the values of `one` (environment) or even `two` qualitative variables

```
p <- ggplot(dat) +
  aes(x = oil, y = yield, color=env, size=protein) +
  geom_point()

p <- p +
  labs(title = "Soybeans, yield and oil in Australia",
       subtitle = "Period 1970-1971",
       caption = "Data: agridat::Australia.soybean",
       x = "Oil (%)",
       y = "Yield (metric tons/hectare)"
       )

p + facet_grid(. ~ env)
p
```

The ggplot2: scatter plot



The ggplot2: Practical

10:00 minutes

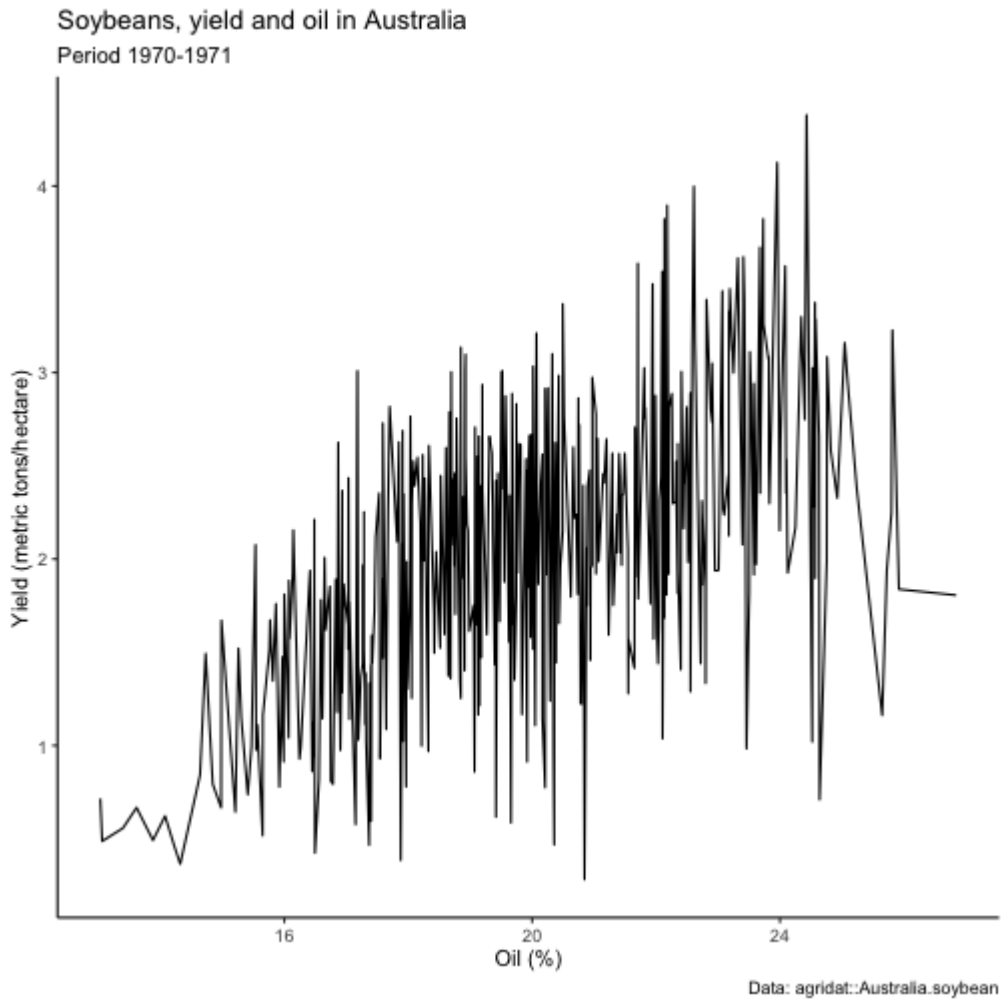
- Use the `australia.soybean` in the `agridat` package
- Visualize the relationship between `oil` in the `x-axis` and `protein` in the `y-axis` by location
  - change the shape of the points (choose one as you see fit)
  - Add a title
  - sub-tile
  - caption
  - change the x-axis and y-axis labels
- Visualize this relationship in a plot with different panels (the number of panels is the number of locations)

The ggplot2: line plot

`Line plots`, particularly useful in time series, can be created by using `geom_line()`

```
p <- ggplot(dat) +
  aes(x = oil, y = yield) +
  geom_line()

p <- p +
  labs(title = "Soybeans, yield and oil in Australia",
       subtitle = "Period 1970-1971",
       caption = "Data: agridat::Australia.soybean",
       x = "Oil (%)",
       y = "Yield (metric tons/hectare)"
       )
p <- p + theme_classic()
p
```

The ggplot2: line plot

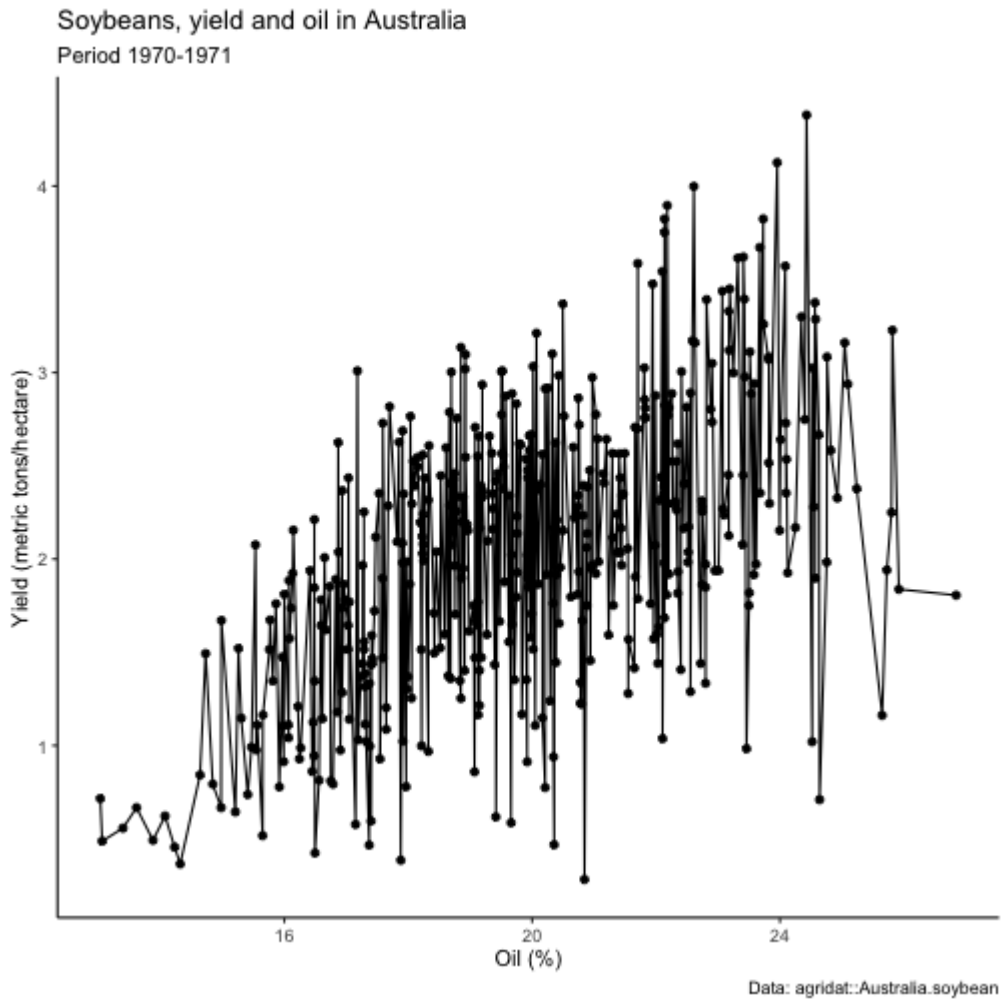


The ggplot2: line plot

We can add a `line` to a `scatter plot` by simply adding a layer to the initial scatter plot

```
p <- ggplot(dat) +
  aes(x = oil, y = yield) +
  geom_point() +
  geom_line()

p <- p +
  labs(title = "Soybeans, yield and oil in Australia",
       subtitle = "Period 1970-1971",
       caption = "Data: agridat::Australia.soybean",
       x = "Oil (%)",
       y = "Yield (metric tons/hectare)"
       )
p <- p + theme_classic()
p
```

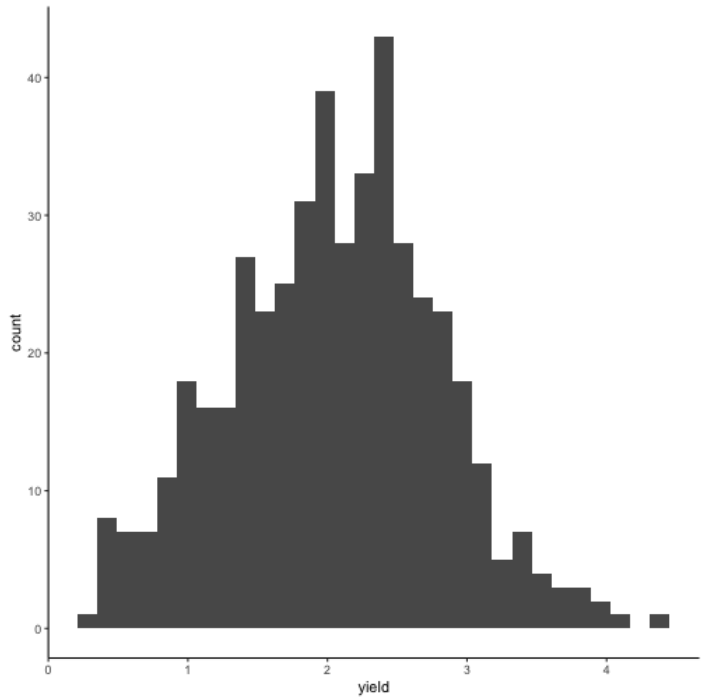
The ggplot2: line plot



The ggplot2: histogram

A histogram can be plotted using `geom_histogram()`

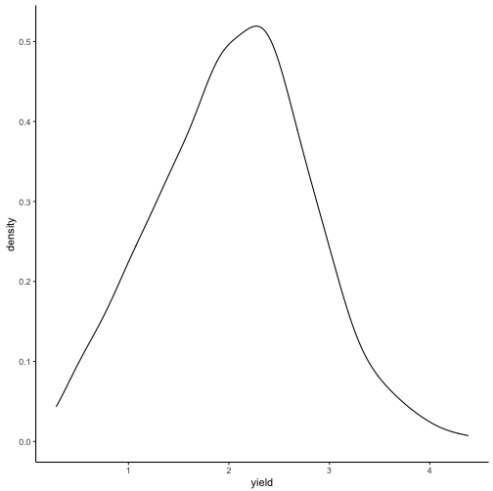
```
p <- ggplot(dat) +
  aes(x =  yield) +
  geom_histogram()
p <- p + theme_classic()
p
```



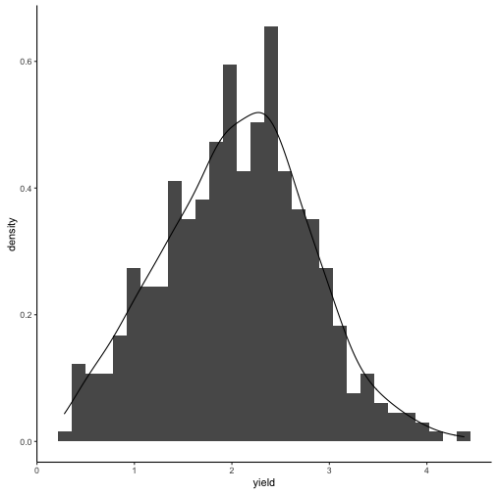
The ggplot2: density plot

`Density plots` can be created using `geom_density()`

```
p <- ggplot(dat) +
  aes(x=yield) +
  geom_density()
p <- p + theme_classic()

p
```




```
p <- ggplot(dat) +
  aes(x=yield, y=..density..)+
  geom_histogram() +
  geom_density()
p <- p + theme_classic()
p
```

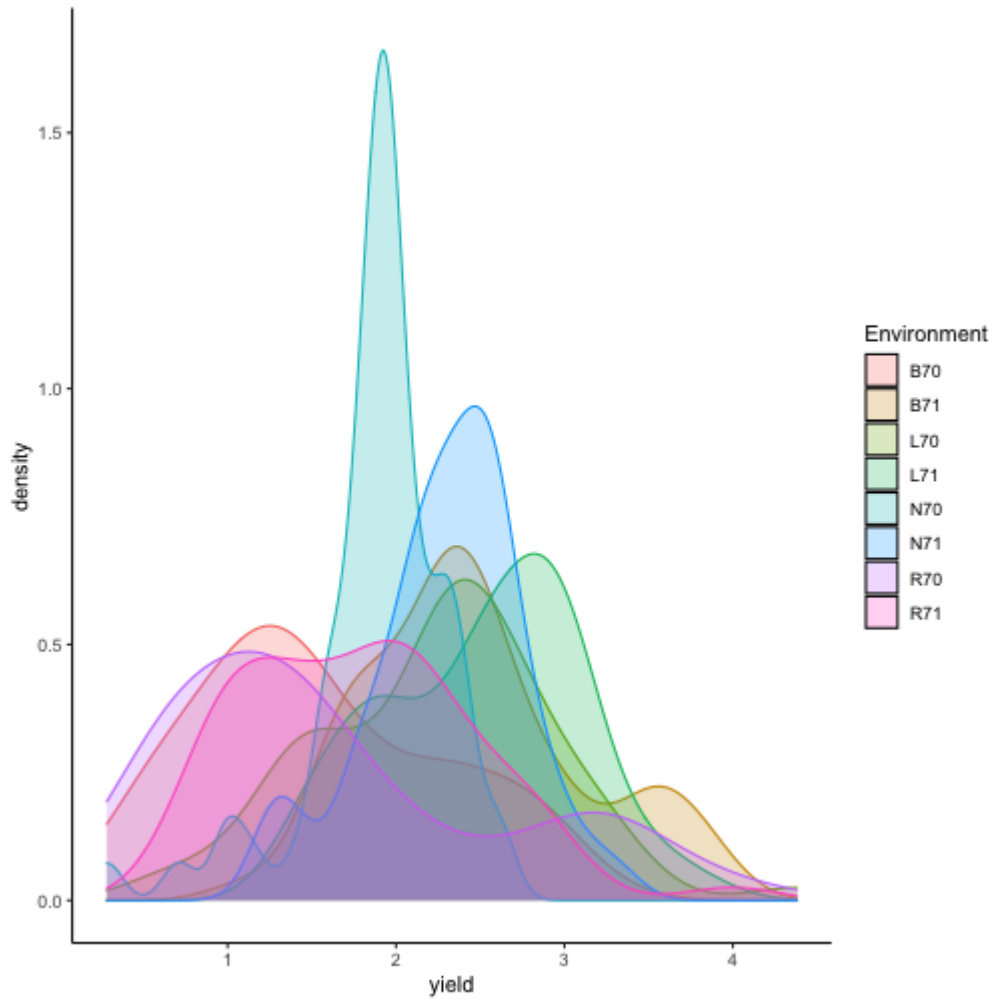


The ggplot2: density plot

Can superimpose several densities by environment

```
p <- ggplot(dat) +
  aes(x=yield, color=env, fill=env) +
  geom_density(alpha = 0.25)
p <- p + scale_fill_discrete(name="Environment")
p <- p + guides(color=FALSE)
p <- p + theme_classic()
p
```

The ggplot2: density plot

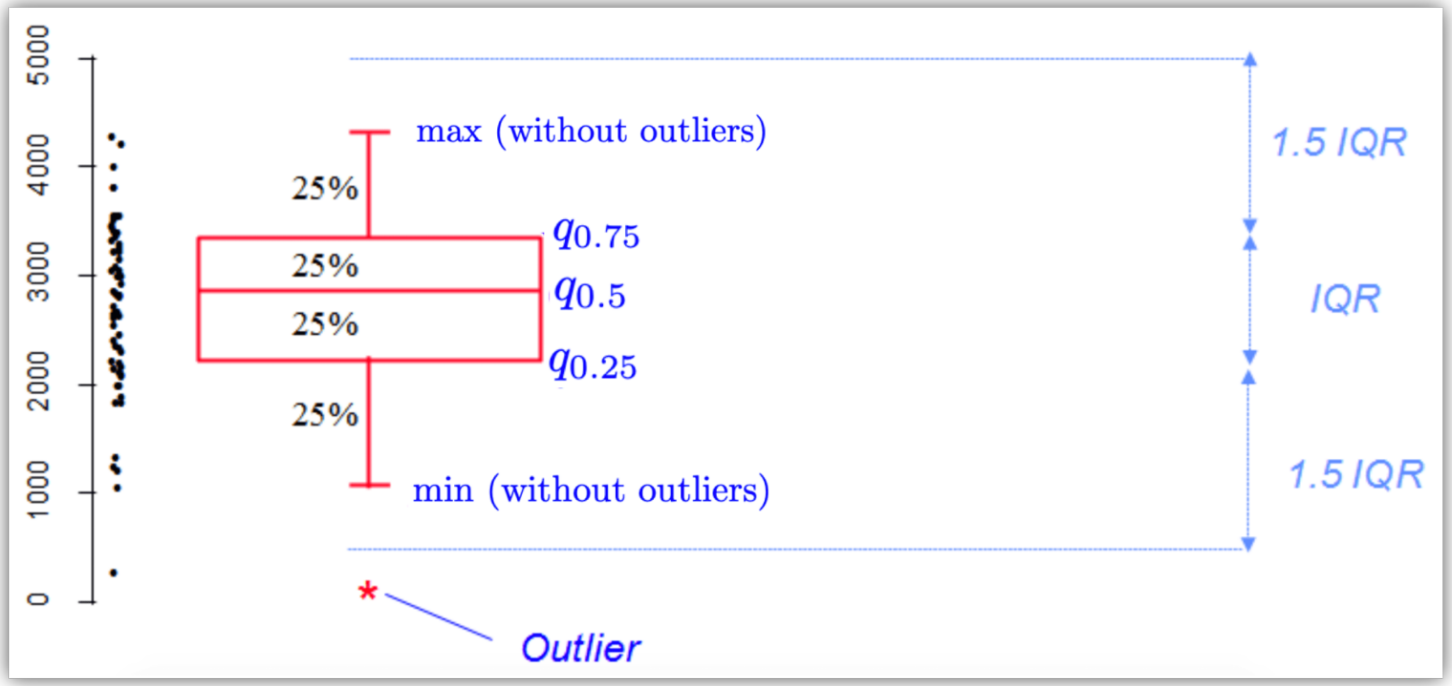


The ggplot2: boxplot

- A boxplot can be plotted using `geom_boxplot()`
- A boxplot graphically represents the distribution of a quantitative variable by visually displaying five common location summary (**minimum**, **median**, **first/third quartiles** and **maximum**) and any observation that was classified as a suspected outlier using the interquartile range (IQR) criterion
- All observations
    - above q0.75 + 1.5· IQR
    - below q0.25−1.5· IQR

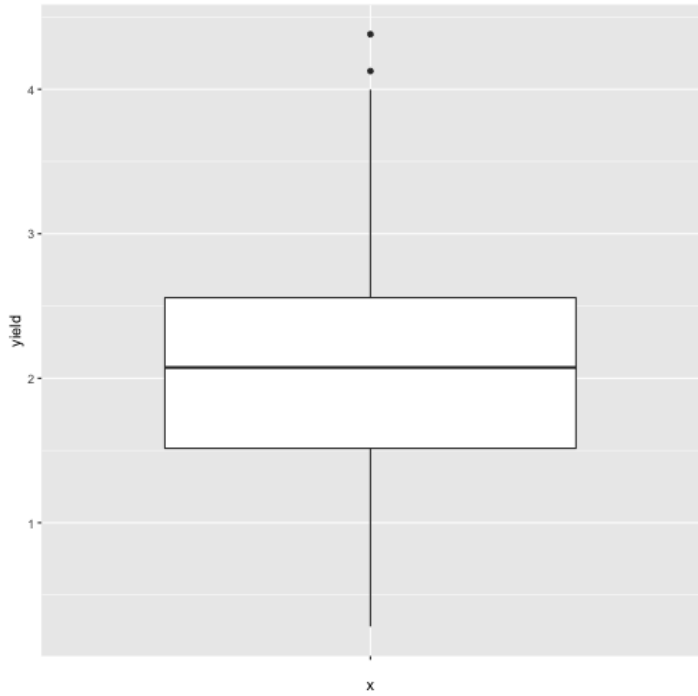
  where q0.25 and q0.75 correspond to first and third quartile respectively are considered as potential outliers by **R**.
- The minimum and maximum in the boxplot are represented without these suspected outliers

The ggplot2: boxplot



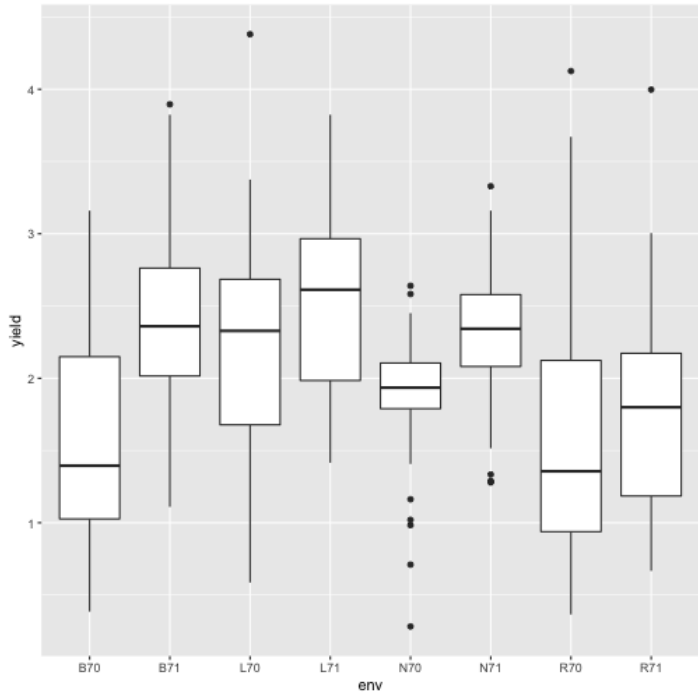
The ggplot2: boxplot

```
p <- ggplot(dat) +
  aes(x = "", y = yield) +
  geom_boxplot()

p
```



The ggplot2: boxplot

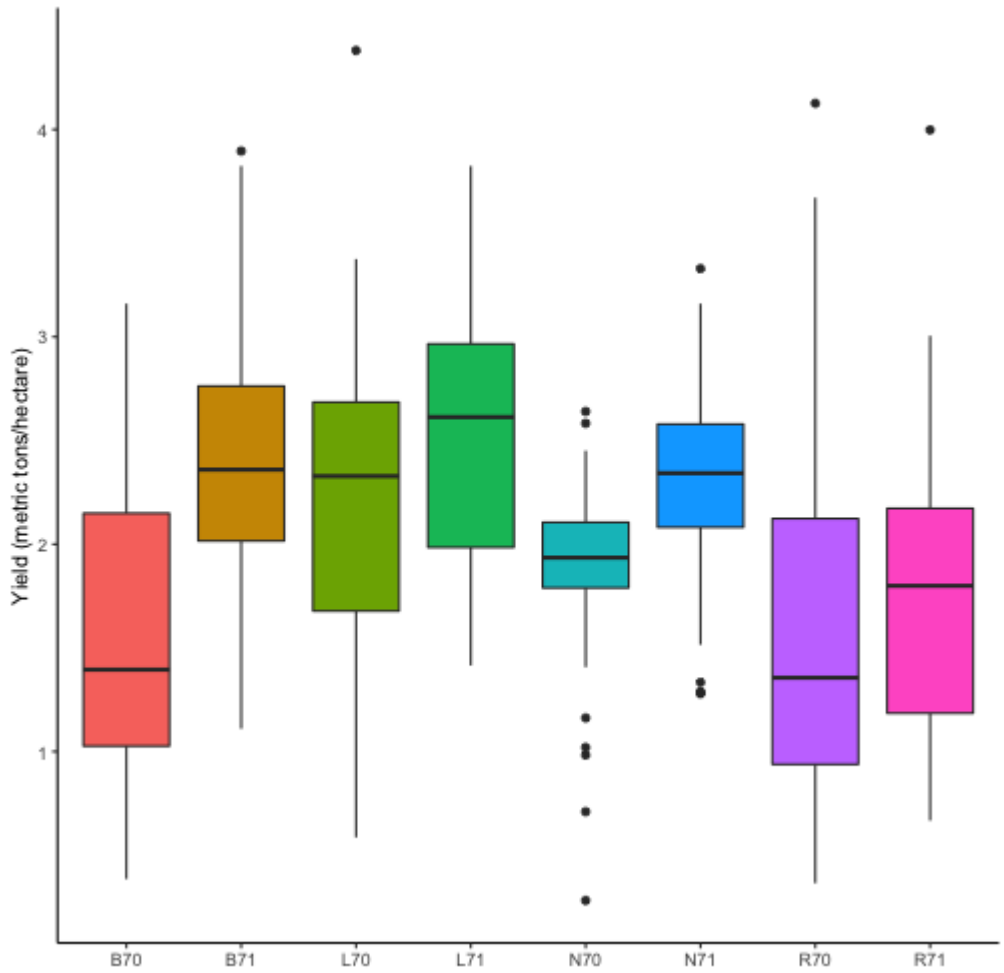
```
p <- ggplot(dat) +
  aes(x = env, y = yield) +
  geom_boxplot()

p
```



The ggplot2: boxplot

```
p <- ggplot(dat) +
  aes(x = env, y = yield, fill = env) +
  geom_boxplot()
p <- p + labs(x = "", y = "Yield (metric tons/hectare)")
p <- p + guides(fill=FALSE)
p <- p + theme_classic()
p
```

The ggplot2: boxplot

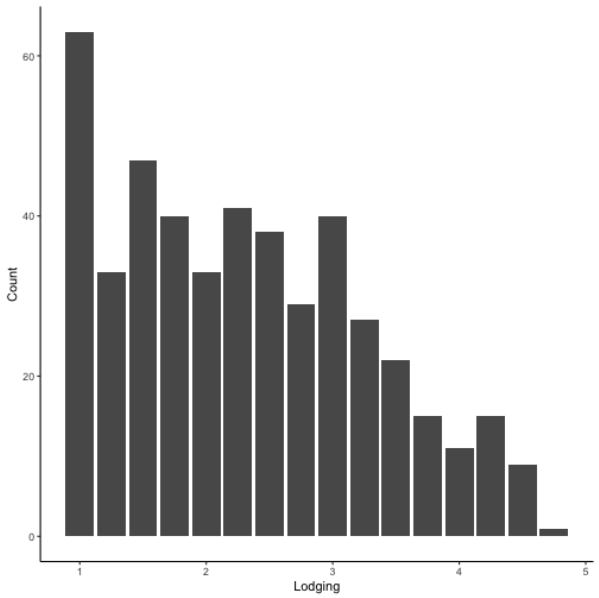


The ggplot2: barplot

- A barplot can be plotted using `geom_bar()`
- A barplot is a tool to visualize the distribution of a qualitative variable (factor). Let's look at the variables in the dataset again
    - **env** (environment), 8 levels, location-year
    - **loc** (location)
    - **year**
    - **gen** (genotype) of soybeans,
    - **yield**, metric tons/hectare
    - **height**, meters
    - **lodging**
    - **size seed**, millimetres
    - **protein**, percentage
    - **oil**, percentage

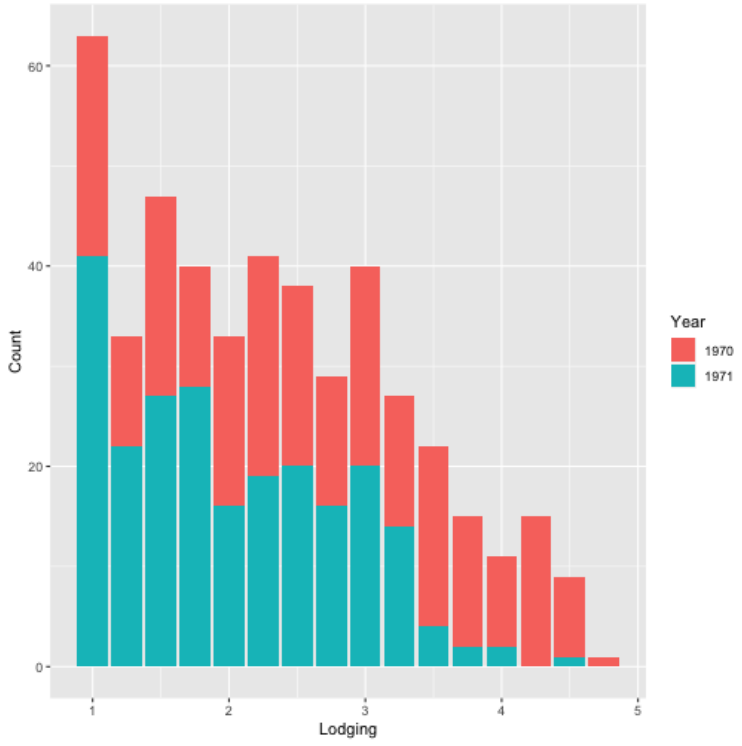
The ggplot2: barplot

```
p <- ggplot(dat) +
  aes(x = lodging) +
  geom_bar()
p <- p + labs(x = "Lodging", y = "Count")
p <- p + theme_classic()
p
```



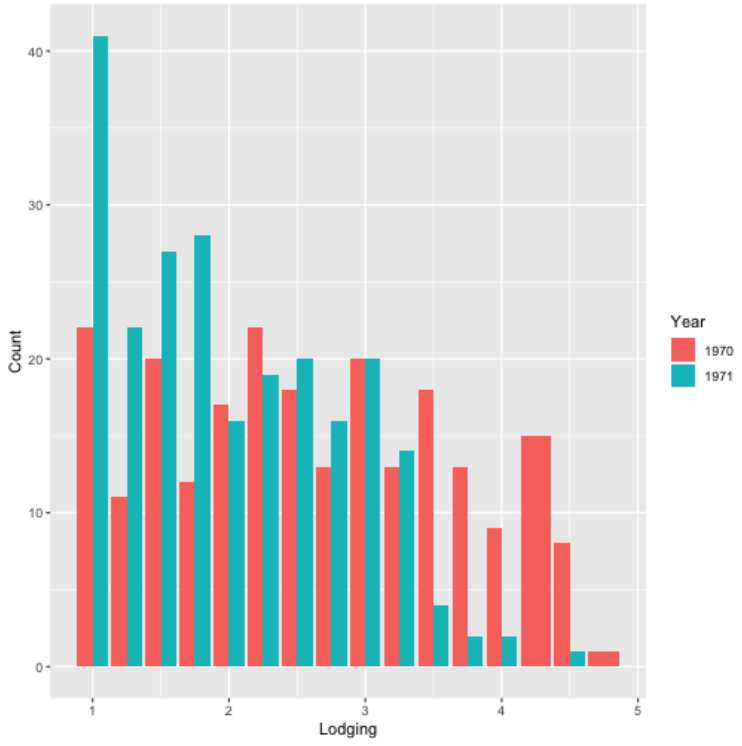
The ggplot2: barplot

```
p <- ggplot(dat) +
  aes(x = lodging, fill=year) +
  geom_bar()
p <- p + labs(x = "Lodging", y = "Count")
p + scale_fill_discrete(name="Year")
```



The ggplot2: barplot

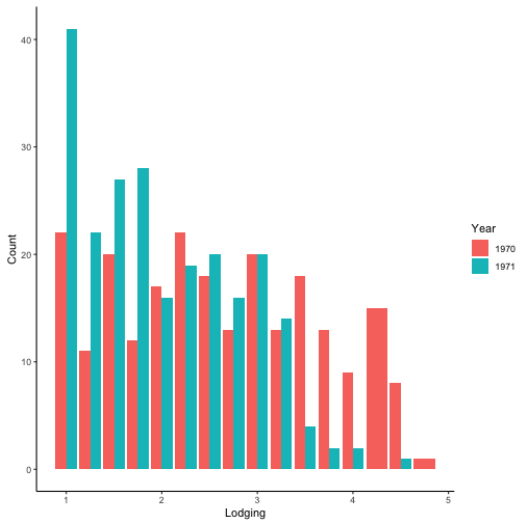
```
p <- ggplot(dat) +
  aes(x = lodging, fill=year) +
  geom_bar(position="dodge")
p <- p + labs(x = "Lodging", y = "Count")
p + scale_fill_discrete(name="Year")
```



The ggplot2: barplot

The function `ggsave()` will save the most recent plot to disk

```
p <- ggplot(dat) +
  aes(x = lodging, fill=year) +
  geom_bar(position="dodge")
p <- p + labs(x = "Lodging", y = "Count")
p <- p + scale_fill_discrete(name="Year")
p + theme_classic()
ggsave("my-plot.png")
```



The ggplot2: Practical

30:00 minutes

- Use the `ars.earlywhitecorn96` in the `agridat` package
- Visualize the relation between `earth` and `yield` by location. Add a title, sub-tile, caption; change the x-axis and y-axis labels; conclusion?
- Display together the histogram and the density of `yield` into different panels, each representing a location
- Display the boxplot of yield by location using colors

The ggplot2: Practical

30:00 minutes

- Import the `iris.xlsx` into **R**
- Make a summary of the table: calculate and display the mean, standard deviation, variance, number of observations by `Species`
- Visualize the relation between `Sepal.Length` and `Petal.Length` by `Species`
- Add a regression line for each of the `Species`
- Display the boxplot of `Sepal.Width` by `Species` using colors
  - Add a title
  - Change the `x-axis` label
  - Change the `y-axis` label