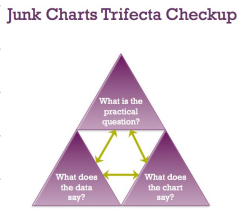

DATA
VISUAL

Week 1

- Trifecta Check up
- DV check list

Data Type:

- Ratio
- Interval
- Nominal
- Ordinal



DV Check-list:

- Text
- Arrangement
- Color
- Lines
- Overall

Check-list Tool: https://datavizchecklist.stephanieevergreen.com/

DV Ethical Principles:

- Beneficence
- Transparency
- Accuracy
- Objectivity
- Respect
- Accountability

Data Integrity

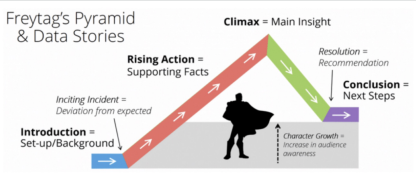
- Permission
- Security
- Consent
- Privacy and sensitive information
- Data quality
- Citation

Basically introduction of what is DV, and how to evaluate quality of a DV. Principles of Ethical and Data Integrity is provided.

Week 2 - Story Telling

DV Storytelling Elements
- Genre
- Structure
- Highlighting
- Transition
- Ordering
- Interactivi

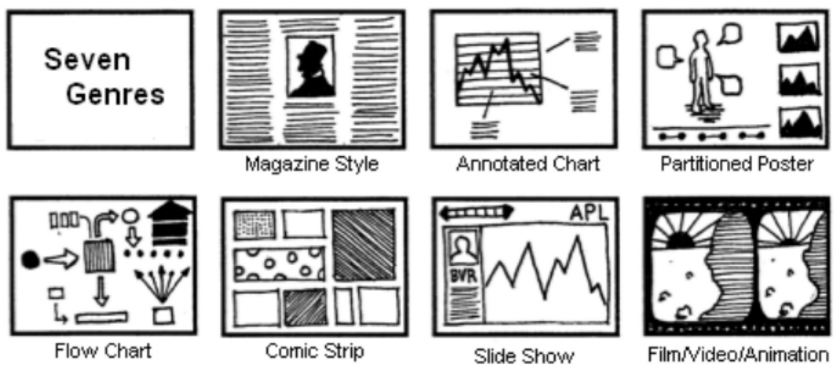
- **Structure:**



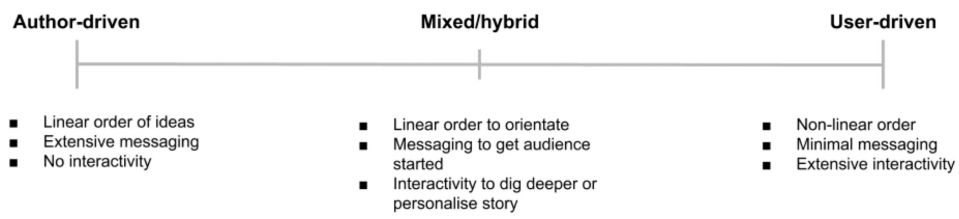
Intro → Supporting Facts → Main Insight → Conclusion

- **Genre:**

7 genres:



- **Approach:**



- **Strategies:**

1 Annotation
2 Visual highlighting
3 Matching content
4 Progress Bars
5 Consistent Visual Platform
6 Multi-messaging

7 Details on demand
8 Timeline slider
9 Tacit tutorial
10 Semantically consistent
11 Markers of interactivity
12 Animated transitions

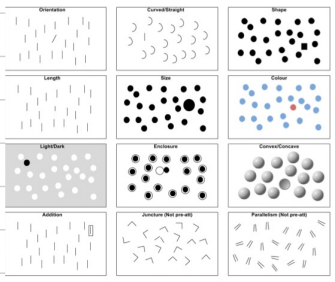
WEEK 3 - Visual Perception and Color

Visual Info Processing System

Colin Ware 3 stage model

Preattentive Processing

visual objects → encode for attention

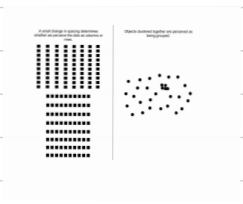


- Orientation
- Curved/straight
- Shape
- Length
- Size
- Colour
- Light/dark
- Enclosure
- Convex/Concave
- Addition

Gestalt Laws → Explain Human recognise visual patterns

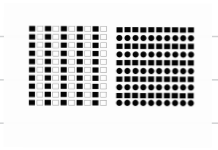
- Proximity
- Similarity
- Connectedness
- Continuity
- Symmetry
- Closure
- Figure Ground Principle
- Common Fate

Proximity



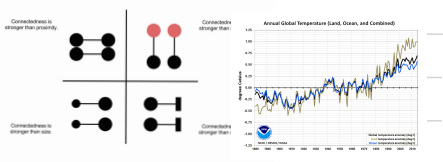
Similarity

eg size, shape, colour



Connectedness

by lines

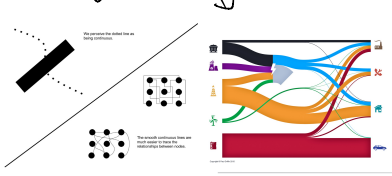


Continuity

smooth & continuous

⇓

easier to perceive



Closure

tend to "fill gap"

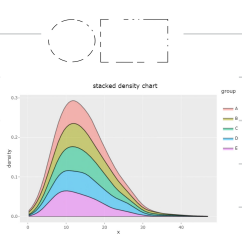
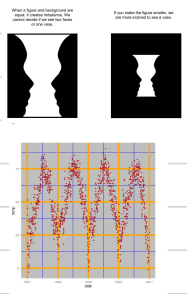


Figure Ground Principle

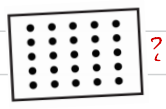
smaller object ⇒ foreground

larger ⇒ background



Common Fate

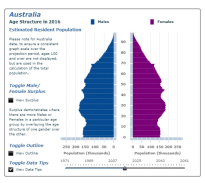
objects move in same direction



Symmetry

tend to group tgt

[ ] { } ( )



Blindness ← change / Inattentional

Visual Variables

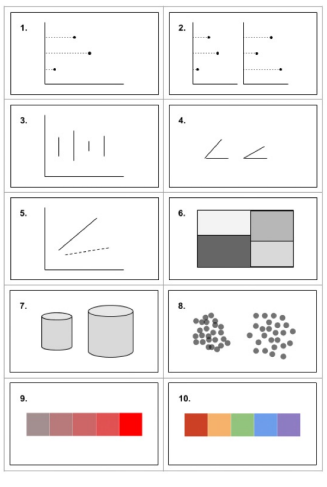
Deconstruct activity

Feature/Object	Example	Notes	Nominal	Ordinal	Quantitative
Position		Powerful and accurate			✓
Length		Powerful and accurate			✓
Size		Less accurate, but still useful			✓
Angle/Slope		Less accurate, but still useful			✓
Shape		Can only use a few different shapes.	✓		
Colour - Hue		Can only use a few different hues.	✓		
Colour - Sequential		Can only use a few different ordered levels.		✓	
Colour - Diverging		Can only use a few different ordered levels.		✓	
Colour - Continuous		Less accurate than other quantitative encodings			✓
Texture		Bygone feature, but may still have applications.	✓		
Line - Type		Can only use a very limited number of types.	✓		
Line - Weight/Boldness		Not very accurate, but still useful.			✓

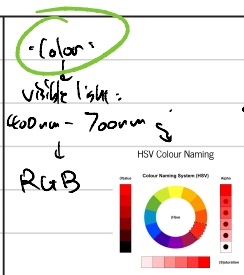
Visual Comparison Accuracy

Visual Comparison Accuracy
(Cleveland and McGill, 1985)

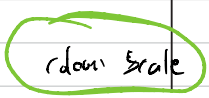
Rank (Most - Least)	Feature
1	Position - common scale
2	Position - same scale , but unaligned
3	Length
4	Angle
5	Slope
6	Area
7	Volume
8	Density
9	Colour saturation
10	Colour hue

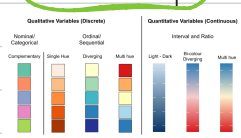


Week 3 - continued



- Select effective colour palettes based on:
  - Sequential
  - Diverging
  - Qualitative data
  - Single or multi-hue
- Colour-blind and print-safe palettes
- Web tool - http://colorbrewer2.org/





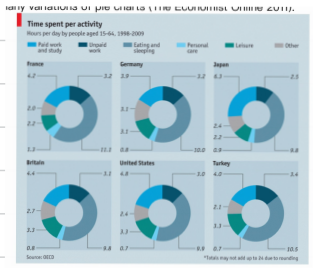
Colour	Positive	Negative
Red	Passion, strength, energy, heat, love	Blood, war, fire, danger, anger, aggressive
Green	Nature, spring, fertility, safety, environment	Inexperience, decay, envy, misfortune
Yellow	Sun, summer, gold, harvest, optimism	Cowardice, treason, hazard, illness, folly
Blue	Sky, sea, stability, peace, unity, depth	Depression, obscenity, conservatism, passivity
White	Snow, purity, peace, cleanliness, innocence	Cold, clinical, surrender, sterility, death, banality
Gray	Intelligence, dignity, restraint, maturity	Shadow, concrete, drabness, boredom
Black	Coal, power, formality, depth, solidarity, style	Fear, void, night, secrecy, evil, anonymity

Week 4 - avoiding deception

Issues

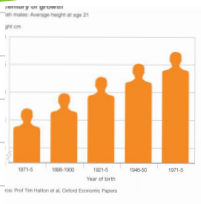
- Issue with Pie chart:
  - Lack Accuracy with Area & Angle vs Position (eg. bar chart)
  - Poor when proportion similar.
  - Rely on Colour
  - limited in num of categories
  - small proportion pies are hard to see

Eg fix deception from pie:

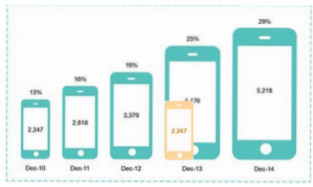


solutions on slides etc.

Truncate Axes



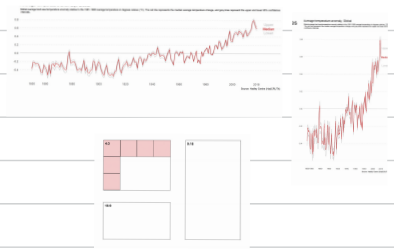
Use Area/size to depict quantity



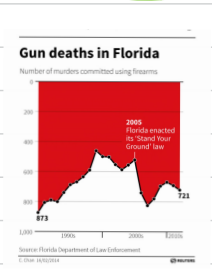
Changing Plot ratio

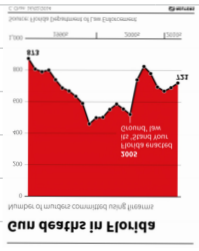
Avoid extreme ratio

Consider effect of dynamic ratio on web browser

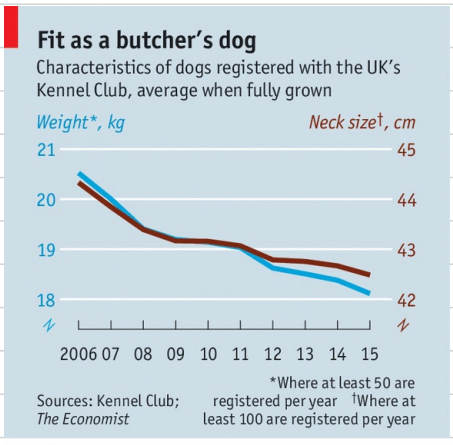


Ignore convention

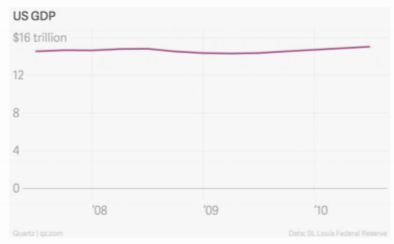




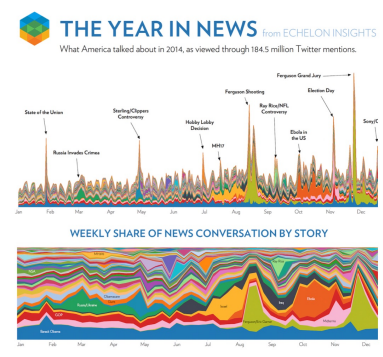
Dual Axes?



Poor scaling methods



Visual bombardment



Week 5 - Grammar &vocabulary

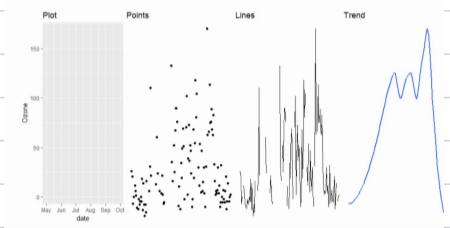
• Layered Grammar of Graphics

Wickham proposed: a graphic is a series of layers consists of:

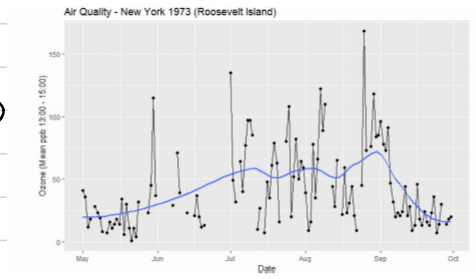
- a default dataset and set of mappings from variables to aesthetics,
- one or more layers, with each layer having
  - one geometric object
  - one statistical transformation,
  - one position adjustment,
  - and optionally, one dataset and set of aesthetic mappings
  - one scale for each aesthetic mapping used
- a coordinate system
- an optional facet specification.

Layers

- Any graphic can be thought of as a series of layers...



- Put them together and we create a graph...



Data

Geometric objects

Represent Data / Statistical Transformation

- Boxes used in boxplots
- Bins used in histograms
- Bars used in barchart
- Points in a scatter plot
- Lines in a line chart

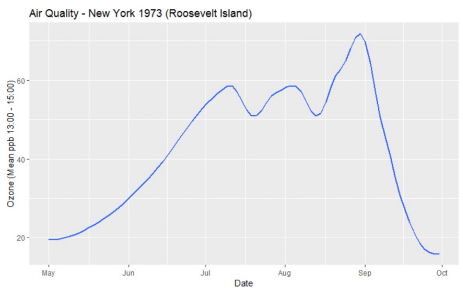
Mapping Aesthetics

- Properties of geometric objects, such as points, lines, colours, and shapes, are referred to as **aesthetics**
- The process of assigning variables from a dataset to aesthetics is known as **mapping**.
- For example in the air quality example x = Date and y = Ozone.
- Points are geometric objects and the position they are drawn on the plot is determined by the mappings.

Statistical Transformation

- Examples of `stats` transformations include the following:
  - quartiles of box plots
  - means
  - error bar/confidence intervals
  - binning in histograms and dot plots
  - tallies, counts, proportions, percentages in bar charts
  - lines of best fit for linear regression.
- Statistical transformations are the reason why statistics is so important for data visualisation.

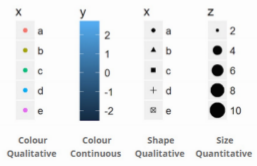
The smoothed trend line in the Air Quality visualisation was estimated using a non-parametric, locally weighted regression model



Scale

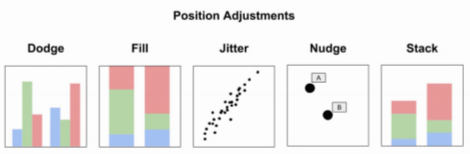
control mapping between a variable & aesthetic

- For example, changing the range of the x axis, or chang the colour of a colour scale.



Position Adjustments

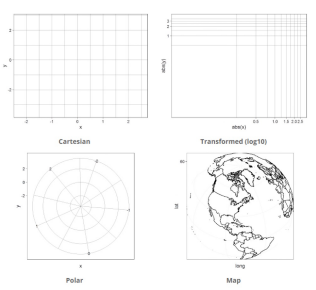
- Position adjustments aim to avoid overlapping elements by either dodging, filling, jittering, nudging or stacking. Layers can incorporate multiple position adjustments.



Coordinate System

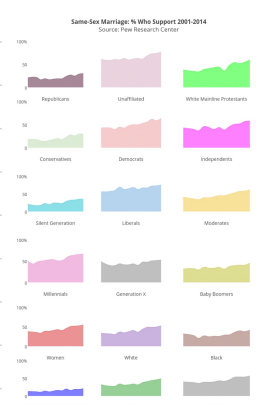
Determine placement of Geometric objects within a plot

- Cartesian
- Transformed
- Polar
- Map



Faceting (latticing / trellising)

Break DV into small multiples



Week 5 - Grammar &vocabulary - continue

ggplot2.

R- Package based on Layered Grammar of Graphics

Powerful grammar for building DV.

Build DV layer by layer

1. Install & Import:

```
install.packages("ggplot2")
library(ggplot2)
```

2. Create ggplot2 object: (define coordinate system & select scales)

```
ggplot() +
  coord_cartesian() +
  scale_x_date(name = "Date") +
  scale_y_continuous(name = "Ozone (Mean ppb 13:00 - 15:00)")
```

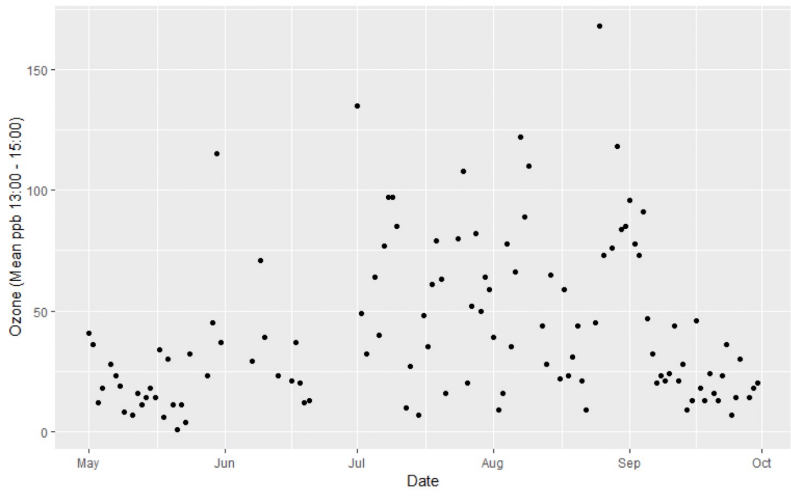
=>

Nothing because we have not defined any layers.

3. Add layer for points

```
ggplot() +
  coord_cartesian() +
  scale_x_date(name = "Date") +
  scale_y_continuous(name = "Ozone (Mean ppb 13:00 - 15:00)") +
  layer(
    data = airquality,
    mapping = aes(x = date, y = Ozone),
    stat = "identity",
    geom = "point",
    position = position_identity()
  )
```

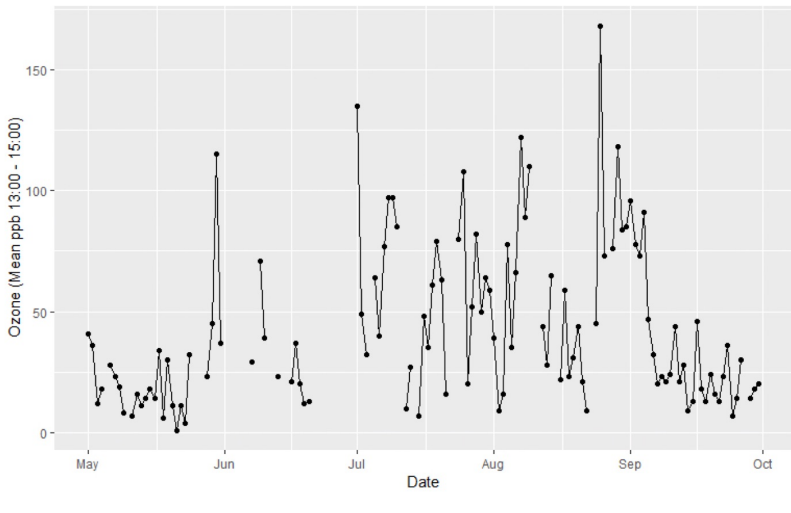
=>



4. Add lines

```
ggplot() +
  coord_cartesian() +
  scale_x_date(name = "Date") +
  scale_y_continuous(name = "Ozone (Mean ppb 13:00 - 15:00)") +
  layer(
    data = airquality,
    mapping = aes(x = date, y = Ozone),
    stat = "identity",
    geom = "point",
    position = position_identity()
  ) +
  layer(
    data = airquality,
    mapping = aes(x = date, y = Ozone),
    stat = "identity",
    geom = "line",
    position = position_identity()
```

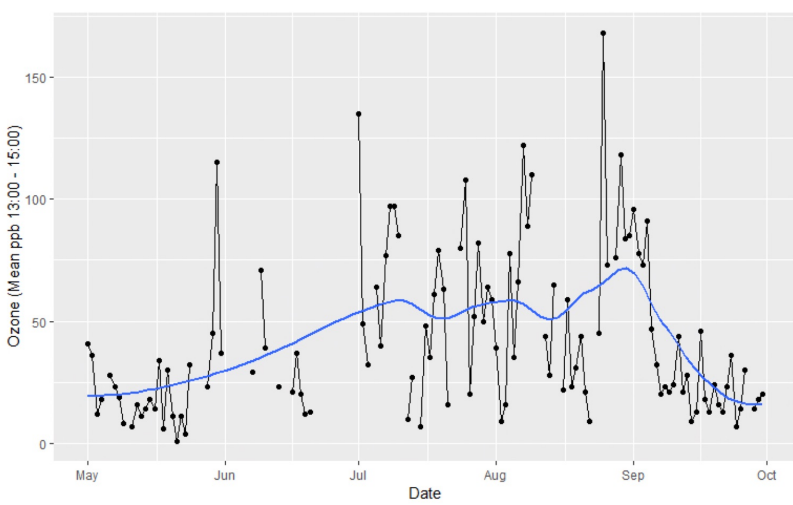
=>



5. Trend lines

```
ggplot() +
  coord_cartesian() +
  scale_x_date(name = "Date") +
  scale_y_continuous(name = "Ozone (Mean ppb 13:00 - 15:00)") +
  layer(
    data = airquality,
    mapping = aes(x = date, y = Ozone),
    stat = "identity",
    geom = "point",
    position = position_identity()
  ) +
  layer(
    data = airquality,
    mapping = aes(x = date, y = Ozone),
    stat = "identity", -> Trend ?
    geom = "line",
    position = position_identity()
```

=>



- The following code will reproduce the same visualisation

```
p <- ggplot(data = airquality, aes(x = date, y = Ozone))
p + geom_point() +
  geom_line(aes(group = 1)) +
  geom_smooth(se = FALSE, span = 0.4) +
  labs(
    title = "Air Quality - New York 1973 (Roosevelt Island)",
    x = "Date",
    y = "Ozone (Mean ppb 13:00 - 15:00)"
    )
```

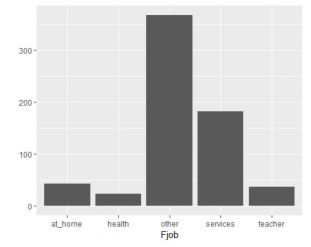
Week 5 - Grammar &vocabulary
- continue,

Read Data:

```
student <- read.csv("../data/Student_alcohol_survey.csv")
```

qplot(): Quick build basic D.V.

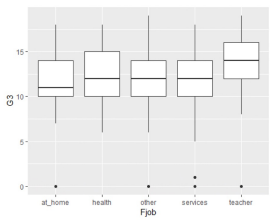
```
qplot(x = Fjob, data = student, geom = "bar")
```



Box plot ⇒

- Box plot:

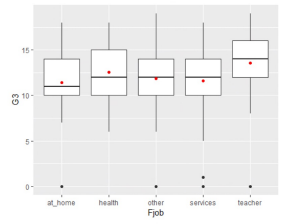
```
qplot(x = Fjob, y = G3, data = student, geom = "boxplot")
```



⇒

- Add some markers for the mean:

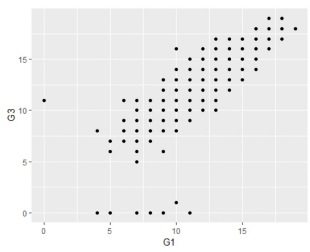
```
qplot(x = Fjob, y = G3, data = student ,geom = "boxplot") +
  stat_summary(fun.y = mean, colour = "red", geom = "point")
```



Scatter plot ⇒

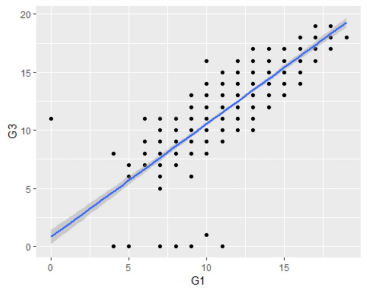
- A basic scatter plot:

```
qplot(x = G1, y = G3, data = student, geom = "point")
```



⇒

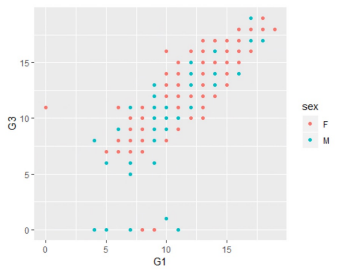
```
qplot(x = G1, y = G3, data = student, geom = "point") +
  geom_smooth(method = "lm")
```



⇓

- A scatter plot with a colour mapped to sex:

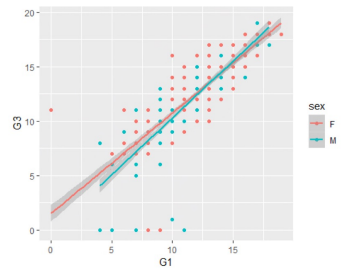
```
qplot(x = G1, y = G3, data = student, colour = sex, geom = "point")
```



⇒

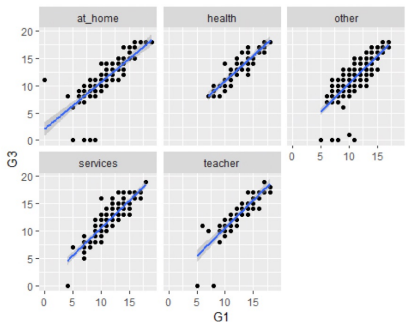
Add individual lines of best fit for each factor level
to the colour aesthetic:

```
qplot(x = G1, y = G3, data = student, colour = sex, geom = "point
  geom_smooth(method = "lm")
```



Facet scatter plots by mother's job:

```
qplot(x = G1,y = G3, data = student, geom = "point") +
  geom_smooth(method="lm") + facet_wrap(~ Mjob)
```



→ Also called "Latticing"

Week 5 - Grammar &vocabulary - continue

ggplot() : more powerful tool building DV.

1. Define ggplot object

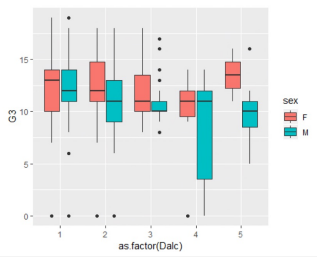
```
p1 <- ggplot(data = student, mapping = aes(x = as.factor(Dalc), y = G3))
```

- `aes` are the aesthetic mappings. Any layers added will map the data to the corresponding aesthetics in the `geom`. For example, a box plot.

2. Map variables : (geom)

We can map additional variables to other scales… Note how we mapped a `fill` aesthetic to `sex`.

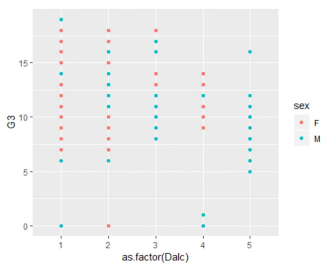
```
p2 <- ggplot(student, aes(x = as.factor(Dalc), y = G3, fill = sex))
p2 + geom_boxplot()
```



- Box plots hide sample size. Use a different `geom` that conveys sample size…

```
p3 <- ggplot(student, aes(x = as.factor(Dalc), y = G3, colour = sex))
p3 + geom_point()
```

=>

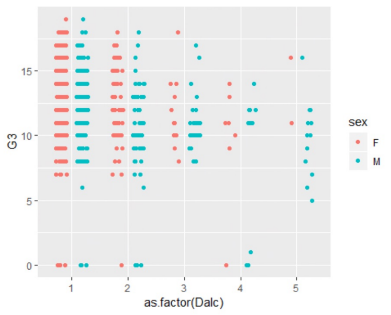


- Note `colour` refers to the outline or solid colour of an object. `fill` refers to the inside colour of a large object, e.g. bar or box.

3. Points are overlapping, Solve. overlapping =>

Use position adjustments to avoid over-plotting

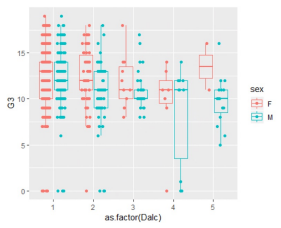
```
p3 + geom_jitter(position = position_jitterdodge())
```



4. Additional geoms →

Note how we use `outlier.shape = NA` to suppress the outliers plotted for the boxplot

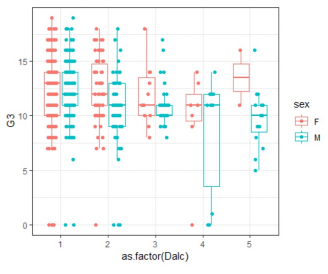
```
p3 + geom_jitter(position = position_jitterdodge()) +
  geom_boxplot(fill = NA, outlier.shape = NA)
```



5. Themes →

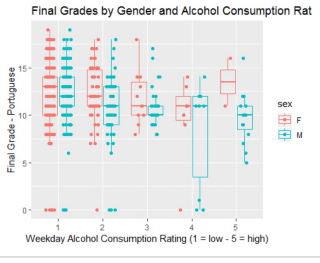
You can change default themes:

```
p3 + geom_jitter(position = position_jitterdodge()) +
  geom_boxplot(fill = NA, outlier.shape = NA) + theme_bw()
```



6. Adding Titles & Labels →

```
p3 + geom_jitter(position = position_jitterdodge()) +
  geom_boxplot(fill = NA, outlier.shape = NA) +
  labs(title = "Final Grades by Gender and Alcohol Consumption Ratings",
       x = "Weekday Alcohol Consumption Rating (1 = low - 5 = high)",
       y = "Final Grade - Portuguese")
```



7. Saving : as image, PDF, or Clipboard

ggsave() -> save as image with sensible defaults to your working directory.

```
ggsave("Box_plot_Grades_Gender_Consumption_01.png",
       width = 18, height = 12, units = "cm")
```

Week 5 - Grammar &vocabulary - continue

Color in R

Color codes
HSV, HSL, RGB, CMYK

Inbuilt R color name

Package e.g. RColorBrewer,

Colourpicker package:

```
install.packages("colourpicker")
```

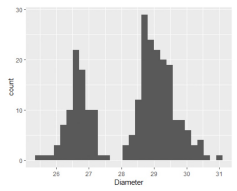
```
library(colourpicker)
```

→ Access Colour Picker in Addins menu in Rstudio

Basic Colour Assignment

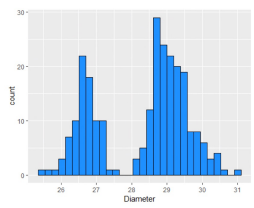
- Produce a histogram showing the distribution of pizza diameter.

```
Pizza <- read.csv("../data/Pizza.csv")
p1 <- ggplot(data = Pizza, aes(x = Diameter))
p1 + geom_histogram()
```



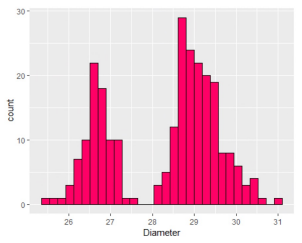
- Change colour using R colour names

```
p1 + geom_histogram(fill = "dodgerblue", colour = "black")
```

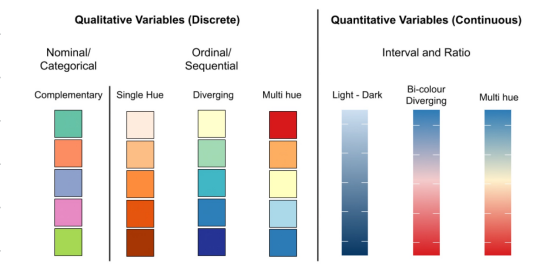


- Change colour using hex codes

```
p1 + geom_histogram(fill = "#ff0066", colour = "#000000")
```



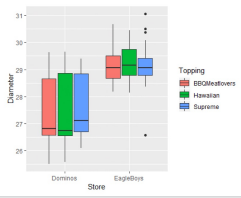
Color Scales



Nominal Color Scale Eg.

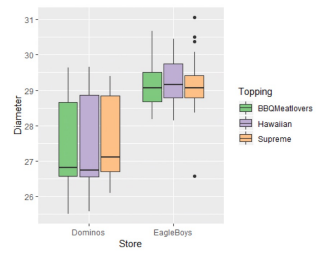
- Side-by-side box plot comparing pizza diameter by Store and Topping

```
p2 <-ggplot(data = Pizza, aes(x = Store, y = Diameter, fill = Topping))
p2 + geom_boxplot()
```



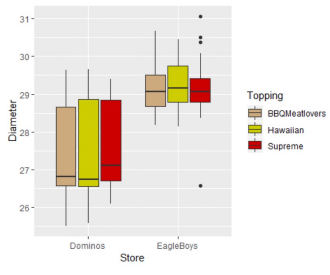
- Change ColourBrewer palette...

```
p2 + geom_boxplot() + scale_fill_brewer(palette = "Accent")
```



- Set manual colour scale...

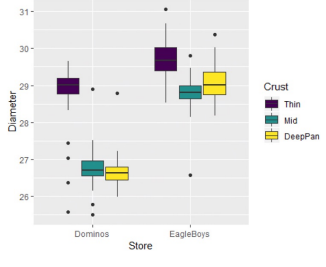
```
p2 + geom_boxplot() + scale_fill_manual(
  values = c("burlywood3","yellow3","red3"))
```



Ordinal Color Scale e.g.

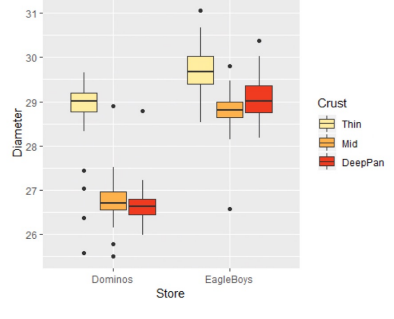
- Boxplot comparing pizza diameter by Store and Crust.

```
Pizza$Crust<-factor(Pizza$Crust, levels = c("Thin","Mid","DeepPan"),
                    ordered = T)
p3 <-ggplot(data = Pizza, aes(x = Store, y = Diameter, fill = Crust))
p3 + geom_boxplot()
```



- Change `palette()`...

```
p3 + geom_boxplot() + scale_fill_brewer(palette = "YlOrRd")
```

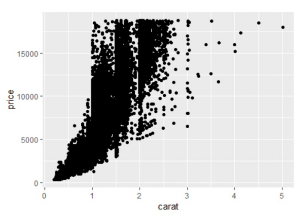


Week 5 - Grammar &vocabulary - continue

Continuous Color Scale

- Explore the relationship between diamond carat and price

```
Diamonds <- read.csv("../data/Diamonds.csv")
p4 <- ggplot(data = Diamonds, aes(x = carat, y = price))
p4 + geom_point()
```

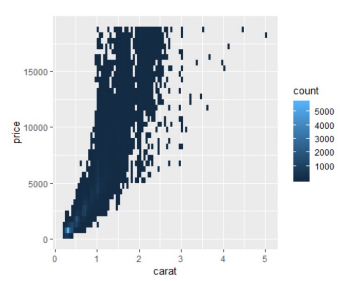


- Hard to see data density as $n = 53940$.

=>

- Use a continuous colour scale to represent data density in a 2d histogram

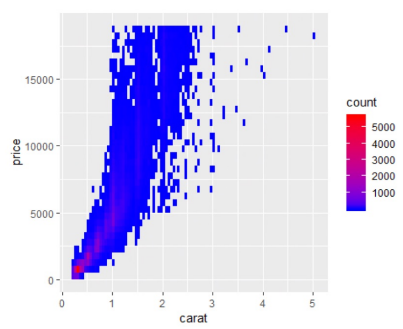
```
p4 + geom_bin2d(binwidth = c(0.05, 500))
```



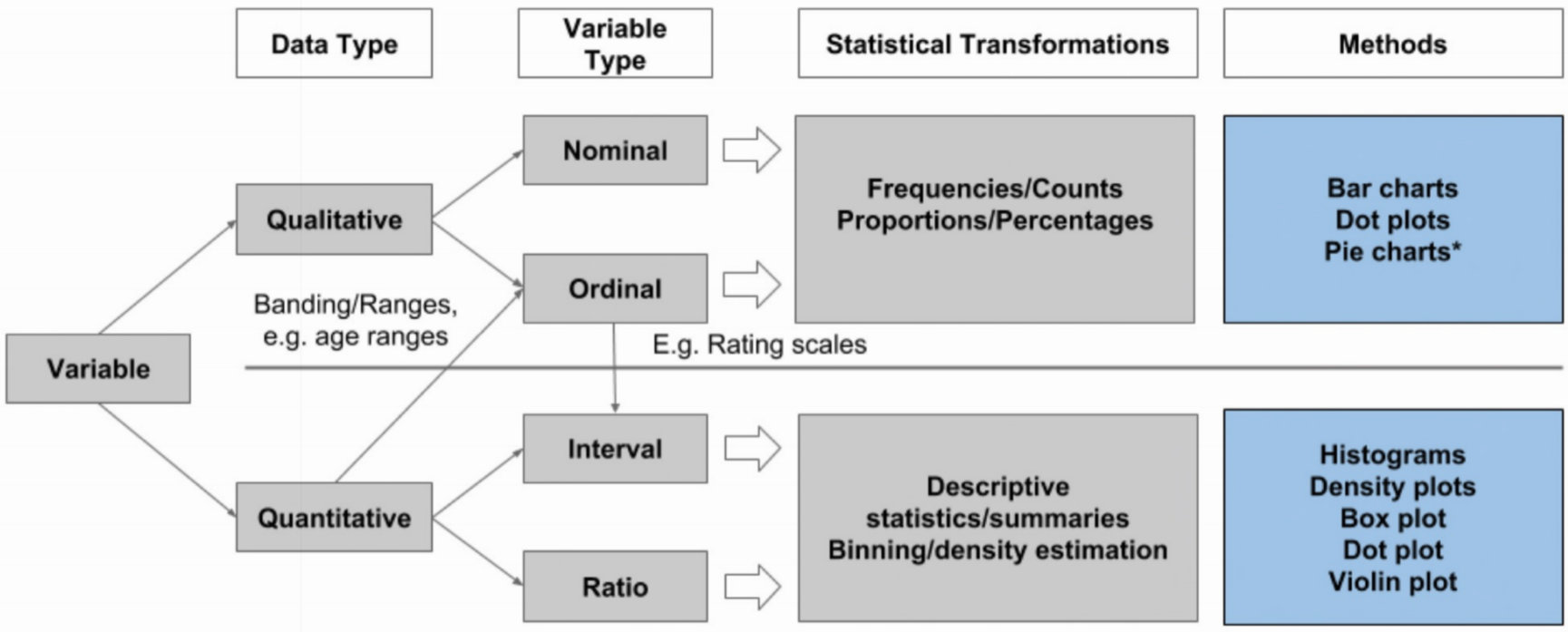
=>

- Change colour gradient...

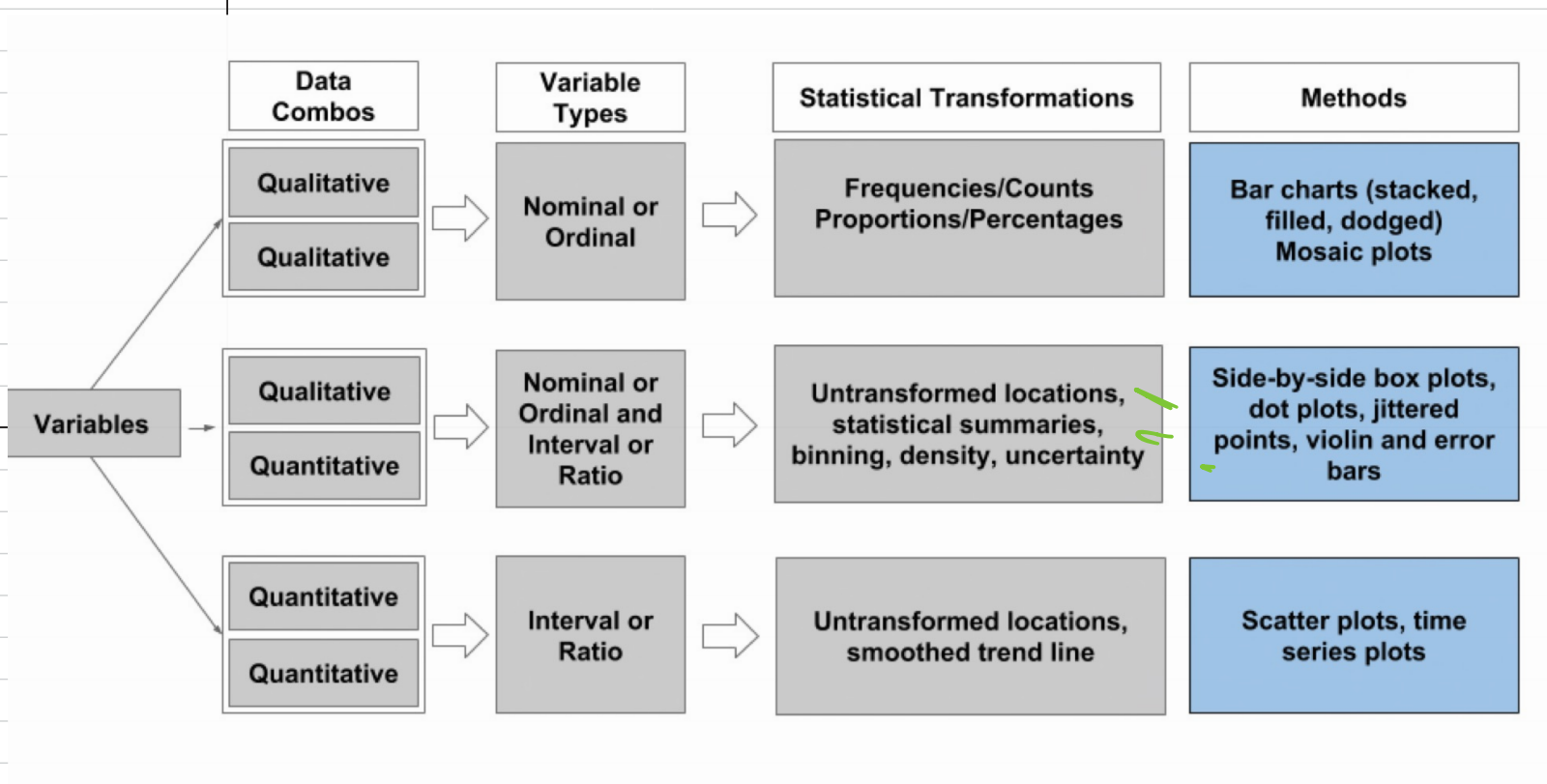
```
p4 + geom_bin2d(binwidth = c(0.05, 500)) +
  scale_fill_gradient(low="blue", high="red")
```



Common Univariate Methods



Common Bivariate Methods



Week 6 - Multivariate Strategies

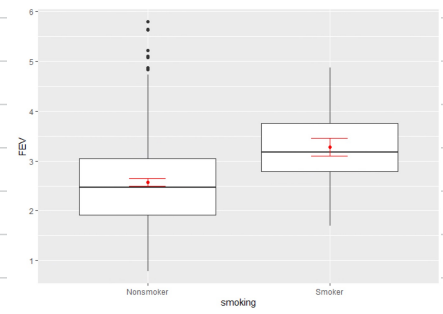
- Mapping Additional Aesthetics
- Faceting
- 3D Scatter.

- Force Expiratory Volume Data:
  - Do children or adolescent smokers tend to have lower FEV readings than non-smokers?

- Bivariate Thinking:
  - Let's start with a side-by-side box plot with mean and error bars (95% CI)

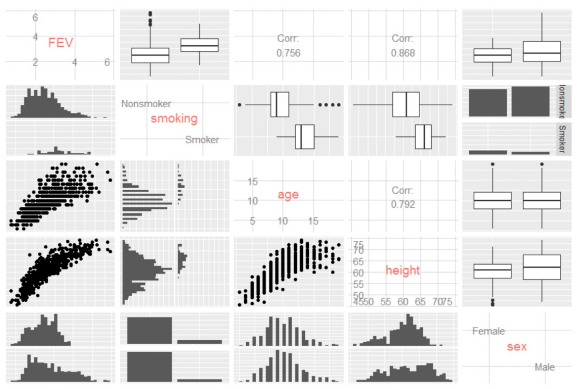
```
FEV = read.csv("../data/FEV.csv")
p1 <- ggplot(data = FEV, aes(x = smoking, y = FEV))
p1 + geom_boxplot() + stat_summary(fun.y = "mean", geom = "p
                                   colour = "red") +
  stat_summary(fun.data = "mean_cl_boot", colour = "red",
               geom = "errorbar", width = .2)
```

=>



- Explore Covariation:

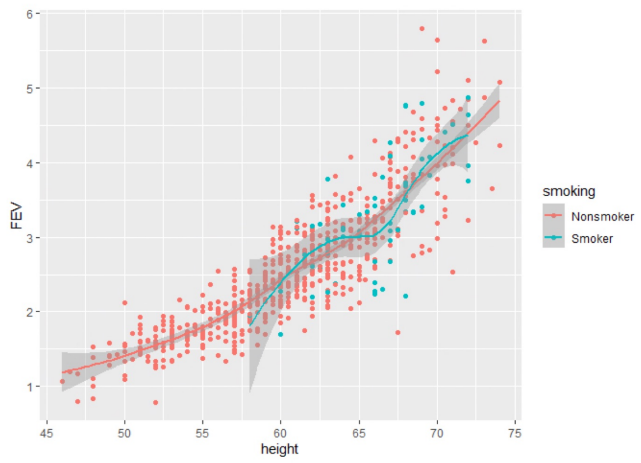
```
library(GGally)
ggpairs(FEV, columns = 1:5,axisLabels = "internal")
```



3 variables

Does Height explain difference in FEV for smokers & non-smoker?

```
p2 <- ggplot(data = FEV, aes(x = height, y = FEV, colour = s
p2 + geom_point() + geom_smooth()
```



Does it also depend on gender?

```
p3 <- ggplot(data = FEV, aes(x = height, y = FEV, colour = s
p3 + geom_point() + geom_smooth() + facet_grid(. ~ sex)
```

- Visualize all in a plot:

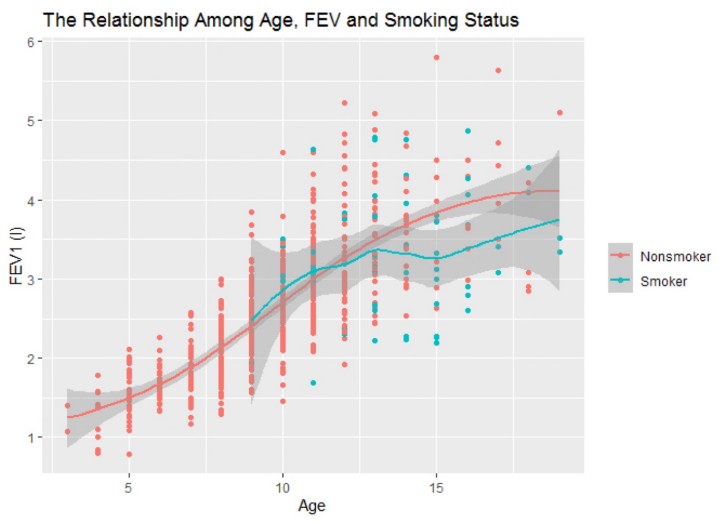
```
library(plotly)
plot_ly(data = FEV, x = ~height, y = ~FEV, z = ~age, color =
        width = 800*1.5, height = 600*1.5) %>%
  add_markers()
```

No pic in slides offline.

- The world is not bivariate.
- Our world is far more interesting and complex.
- The aim of multivariate data visualisation is to see through this complexity

=>

```
p4 <- ggplot(data = FEV, aes(x = age, y = FEV, colour = smok
p4 + geom_point() + geom_smooth() +
  labs(x = "Age", y = "FEV1 (l)",
       title = "The Relationship Among Age, FEV and Smoking
  scale_colour_discrete(name = "")
```

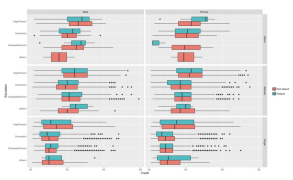


Week 6 - Multivariate Strategies - continue

- Multivariate DVs
- Common methods to do that:
  - Mapping additional Aesthetics
    - ggplot2: `x`, `y` (position), `colour`, `fill`, `shape`, `size`, `weight`, `linetype`, `alpha`...
    - No more than three in one plot (Wickham 2010).
  - Faceting
    - ≤ 2 variables
  - Purpose built.
    - **Purpose built**: Many examples: Sankey diagrams, parallel coordinates, 3D scatter plots and multivariate mosaic plots...
  - Animations
    - **Animation**: Change or transition mapped to a variable, e.g. time (Chapters 8 & 9).

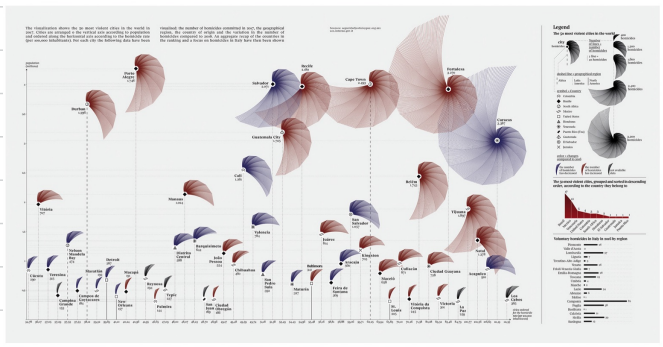
Rule of Thumb ?

Count how many variables are contained in the following visualisation



Multivariate Dilemma:

Violent Cities (Fragapane 2018)



=>

1. Clarify Objective

- To educate the audience about the cities with the highest rates of homicide.
  - Rank cities by homicide rate
  - Show population size
  - Show total homicides
  - Show country/region
  - Change (Increased or decreased from 2016) - no data*

2. Data

- Data was sourced from Citizen Council for Public Security and Criminal Justice AC (2018).
- Download a clean copy **here**.

```
library(readxl)
top50 <- read_excel("../data/top_50_cities_by_homicide_2018.
```

- Original ranks were incorrect

```
top50$Rank = rank(top50$Rate)
```

- Reorder ranks into ascending order

```
library(forcats)
top50$City <- fct_reorder(top50$City, top50$Rank, desc = FAI
```

=>

3. Data - collapse countries

- Convert population unit to millions - easier to plot

```
top50$Population <- top50$Population/1000000
```

- Collapse low tally countries into "other" category

```
table(top50$Country)
```

```
##
##         Brazil       Colombia    El Salvador      Guatemala
##             14              2              1              1
##        Jamaica         Mexico   South Africa  United States
##              1             15              3              5
```

=>

```
top50$Country_2<- fct_collapse(top50$Country,
  Brazil = "Brazil",
  Mexico = "Mexico",
  United_States = "United States",
  Venezuela = "Venezuela",
  South_Africa = "South Africa",
  Other = c("Colombia", "El Salvador","Guatemala","Honduras"
)

table(top50$Country_2)
```

```
##
##         Brazil          Other         Mexico  South_Africa U
##             14              7             15              3
##      Venezuela
##              6
```

4. order&labels levels

```
top50$Country_2 <- factor(top50$Country_2,
                          levels = c("Mexico",
                                     "Brazil",
                                     "Venezuela",
                                     "Other",
                                     "United_States",
                                     "South_Africa"
                                     ),
                          labels = c("Mexico",
                                     "Brazil",
                                     "Venezuela",
                                     "Other*",
                                     "United States",
                                     "South Africa"
                                     ))
table(top50$Country_2)
```

```
##
##         Mexico         Brazil      Venezuela         Other* U
##             15             14              6              7
##   South Africa
##              3
```

Week 6 - Multivariate Strategies - continue

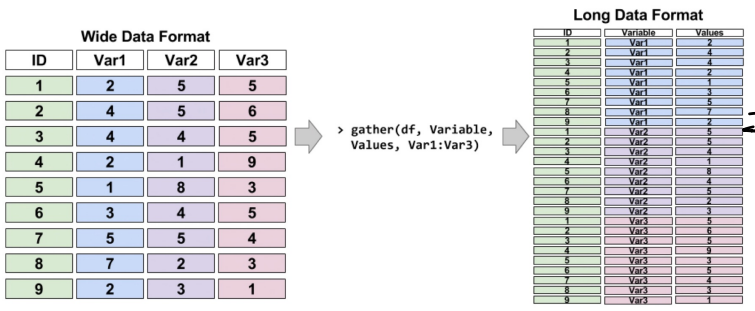
5. Wide to long Data Format

- Data is in a wide format

```
library(data.table)
top50
```

```
## # A tibble: 50 x 8
##      Rank City        Country  Region   Homicides Populatic
##    <dbl> <fct>       <chr>    <chr>        <dbl>      <dbl
##  1    50 Tijuana     Mexico   Latin A~      2640       1.91
##  2    49 Acapulco    Mexico   Latin A~       948       0.85
##  3    48 Caracas     Venezue~ Latin A~      2980       2.98
##  4    47 Ciudad Vi~  Mexico   Latin A~       314       0.36
##  5    46 Ciudad Ju~  Mexico   Latin A~      1251       1.46
##  6    45 Irapuato    Mexico   Latin A~       473       0.58
##  7    44 Ciudad Gu~  Venezue~ Latin A~       645       0.82
##  8    43 Natal       Brazil   Latin A~      1185       1.59
##  9    42 Fortaleza   Brazil   Latin A~      2724       3.94
## 10    41 Ciudad Bo~  Venezue~ Latin A~       264       0.38
## # ... with 40 more rows
```

- Restructure data to long format



- Easy to do using the `tidyr::gather()` function.

```
library(tidyr)
top50_long <- gather(top50, key = "Variable", value = "Value
```

```
## # A tibble: 150 x 7
##      Rank City            Country    Region        Country_
##    <dbl> <fct>           <chr>      <chr>         <fct>
##  1    50 Tijuana         Mexico     Latin America Mexico
##  2    49 Acapulco        Mexico     Latin America Mexico
##  3    48 Caracas         Venezuela  Latin America Venezuel
##  4    47 Ciudad Victoria Mexico     Latin America Mexico
##  5    46 Ciudad Juárez   Mexico     Latin America Mexico
##  6    45 Irapuato        Mexico     Latin America Mexico
##  7    44 Ciudad Guayana  Venezuela  Latin America Venezuel
##  8    43 Natal           Brazil     Latin America Brazil
##  9    42 Fortaleza       Brazil     Latin America Brazil
## 10    41 Ciudad Bolívar  Venezuela  Latin America Venezuel
## # ... with 140 more rows
```

6. Add Rank labels & Variable Labels

- Add a new rank label variable in order to label a county's rank on each variable.

```
top50_long$Rank_labels <- c(rank(-top50$Homicides,ties.metho
                            rank(-top50$Population,,ties.met
                            rank(-top50$Rate,ties.method = "
```

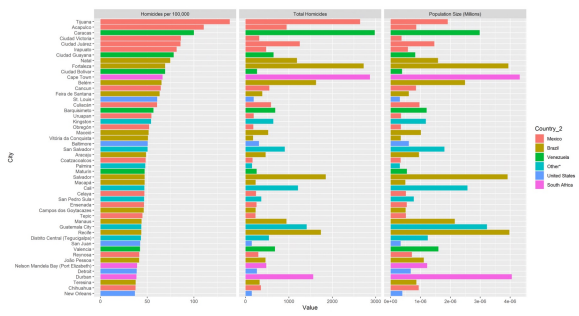
- Clean up factor labels for visualisation.

```
top50_long$Variable <- factor(top50_long$Variable,
                              levels = c("Rate","Homicides",
                              labels = c("Homicides per 100,
                                         "Total Homicides",
                                         "Population Size (M
```

7. Basic Visualisations

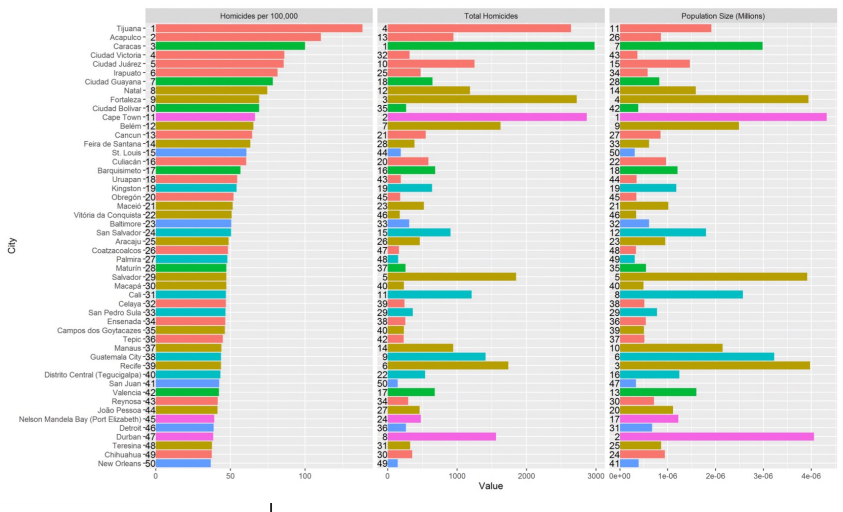
- Make use of facets to align, but isolate each variable.

```
p <- ggplot(data = top50_long,
            aes(x = City, y = Value, fill = Country_2)) +
  geom_bar(stat = "identity") + coord_flip() +
  facet_grid(.~Variable, scales = "free")
```



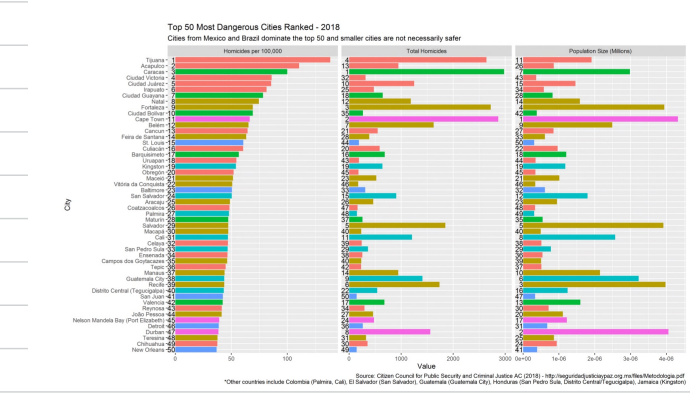
8. Add rank labels

```
p <- p +
  geom_text(aes(label = Rank_labels, y = 0),
            fill = "gray", hjust = "top", family="Georgia")
```



9. Add Details

```
p <- p +
  labs(title = "Top 50 Most Dangerous Cities Ranked - 2018",
       subtitle  = "Cities from Mexico and Brazil dominate t
       caption = "Source: Citizen Council for Public Securit
       *Other countries include Colombia (Palmira, Cali), El
```



10. Adjust Color, Fonts & Clean up

- We can closely replicate a colour theme by obtaining colours from the original
  - background - #EFF1F0
  - red - #89141C
  - blue - #29265B
- Use **coolors** website to create a colour palette.

```
background <- "#EFF1F0"

pal <- c("#89141cff",
         "#b16264ff",
         "#d8b0acff",
         "#d8d0c1ff",
         "#3a506bff",
         "#29265bff"
)
```

- The original data visualisation used a font similar to Georgia.
- To access a greater range of fonts for `ggplot2` use the `extrafont` package.

```
install.packages("extrafont")
library(extrafont)
font_import()
```

Putting this all together...

```
p <- p +
  theme_gray() +
  scale_fill_manual(values = pal) +
  theme(plot.background = element_rect(fill = background),
        panel.background = element_rect(fill = background),
        legend.background = element_rect(fill = background),
        text=element_text(family="Georgia"),
        title = element_text(face = "bold"),
        axis.title.x = element_blank(),
        axis.title.y = element_blank(),
        legend.title = element_blank())
```

- Best viewed full-screen