Data 550: Data Visualization I

Lecture 1: Introduction

Dr. Irene Vrbik
University of British Columbia Okanagan

https://github.com/ubco-mds-2022/Data-550

Outline

Today, we'll look into:

- What is vis?
- What are some plotting tools?
- What is the focus of this course, and what is expected of students overall?

https://github.com/ubco-mds-2022/Data-550

Learning Outcomes

From today's lecture, students are expected to:

- Have a sense of what data visualization is
- Gain a sense of what exploratory data analysis is.
- Learn the basic syntax of R's ggplot2 and Python's Altair

https://github.com/ubco-mds-2022/Data-550

What is data visualization

- At its core, data visualization[1] is about representing numbers with graphical components.

Why not look at the raw numbers?

- Its not easy to learn patterns when looking at written data
- *external representation*[2] allows humans to surpass the limitations of our own internal cognition and memory

1. a.k.a. “vis” or “viz”.
2. a.k.a *external memory* replaces cognition with perception

Why not raw numbers

Can you spot any patterns in this dataset?

x1	x2	x3	x4	y1	y2	y3	y4
10	10	10	8	8.04	9.14	7.46	6.58
8	8	8	8	6.95	8.14	6.77	5.76
13	13	13	8	7.58	8.74	12.74	7.71
9	9	9	8	8.81	8.77	7.11	8.84
11	11	11	8	8.33	9.26	7.81	8.47
14	14	14	8	9.96	8.10	8.84	7.04
6	6	6	8	7.24	6.13	6.08	5.25
4	4	4	19	4.26	3.10	5.39	12.50
12	12	12	8	10.84	9.13	8.15	5.56
7	7	7	8	4.82	7.26	6.42	7.91
5	5	5	8	5.68	4.74	5.73	6.89

https://github.com/ubco-mds-2022/Data-550

Why not summary statistics?

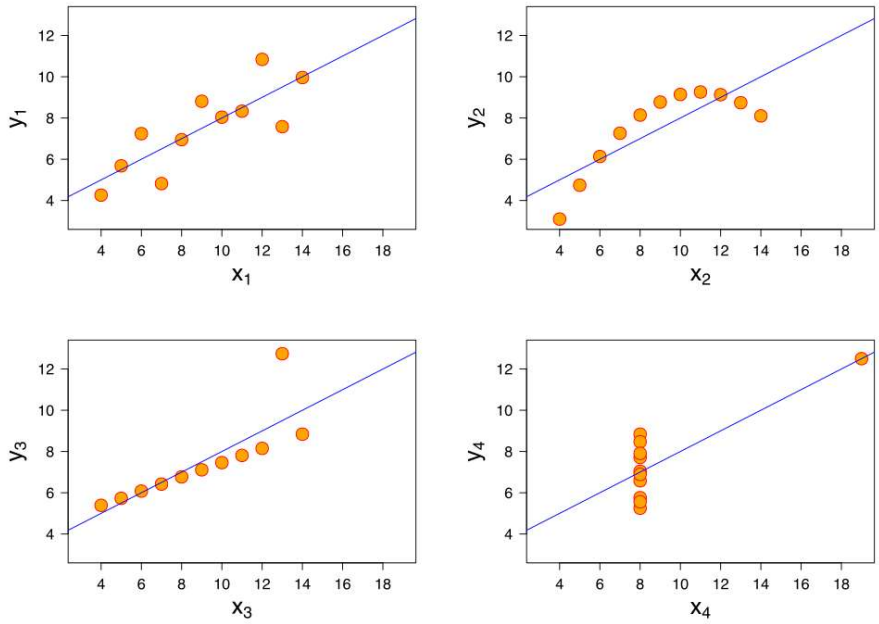
	mean	stdev
x1	9	3.316625
x2	9	3.316625
x3	9	3.316625
x4	9	3.316625

	mean	stdev
y1	7.500909	2.031568
y2	7.500909	2.031657
y3	7.500000	2.030424
y4	7.500909	2.030578

	corr	r2	ols.fit
(x1, y1)	0.8164205	0.6665425	y1 = 3 + 0.5*x1
(x2, y2)	0.8162365	0.6662420	y2 = 3 + 0.5*x2
(x3, y3)	0.8162867	0.6663240	y3 = 3 + 0.5*x3
(x4, y4)	0.8165214	0.6667073	y4 = 3 + 0.5*x4

https://github.com/ubco-mds-2022/Data-550

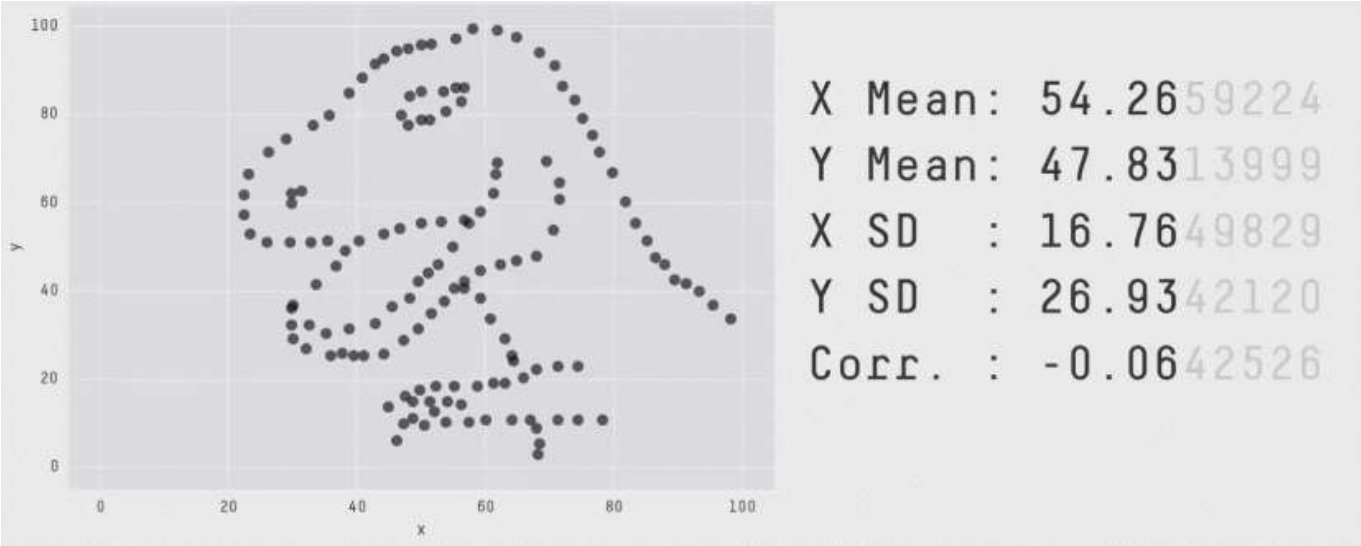
Anscombe's quartet



All four sets are identical when examined using simple summary statistics, but vary considerably when graphed. Photo source: [Wikipedia (Anscombe's quartet, 2023)]

https://github.com/ubco-mds-2022/Data-550

Datasaurus Dozen



GIF created from Matejka and Fitzmaurice (2017) *Same Stats, Different Graphs*

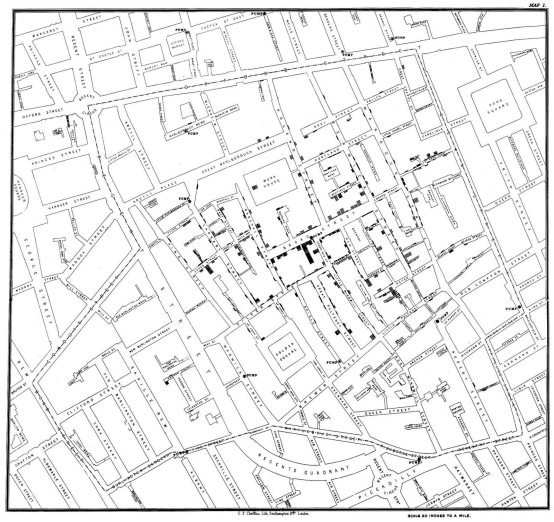
https://github.com/ubco-mds-2022/Data-550

Goals of Data Visualization

- Exploring the unknown: discover patterns, plot anomalies, answer/generate questions, eg what is driving a cholera outbreak?
- Explain the known: “tell a story” to an audience and communicate information in a digestible format, answer a specific question, eg who is the hardest working country in EU nations?
- A stepping stone for developing modes: help to refine/determine parameters and test assumptions

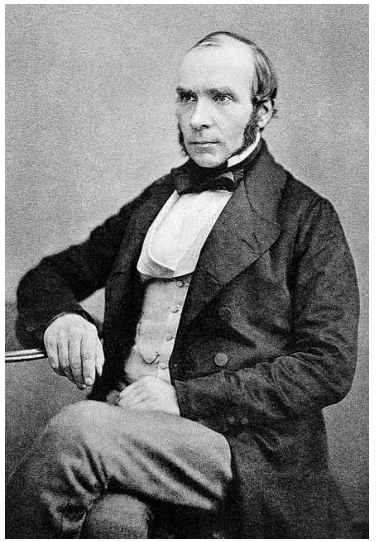
https://github.com/ubco-mds-2022/Data-550

Cholera Outbreak



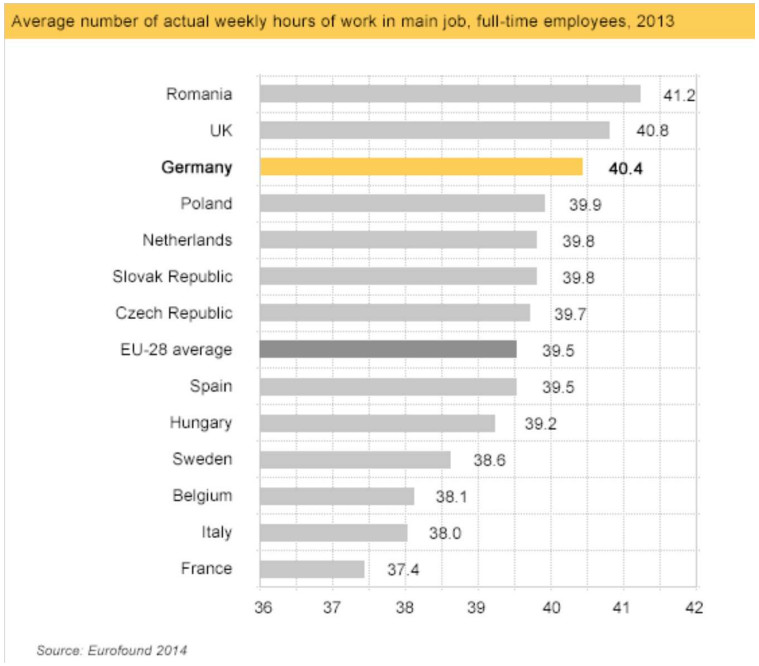
The dot plot used by John Snow to illustrate the cluster of cholera cases around the pump on Broad Street. Image source: [Wikipedia (John Snow, 2023)]

John Snow



https://github.com/ubco-mds-2022/Data-550

EU labour market



Do you think it is fair to say that Germans are more motivated and work more hours than do workers in other EU nations?

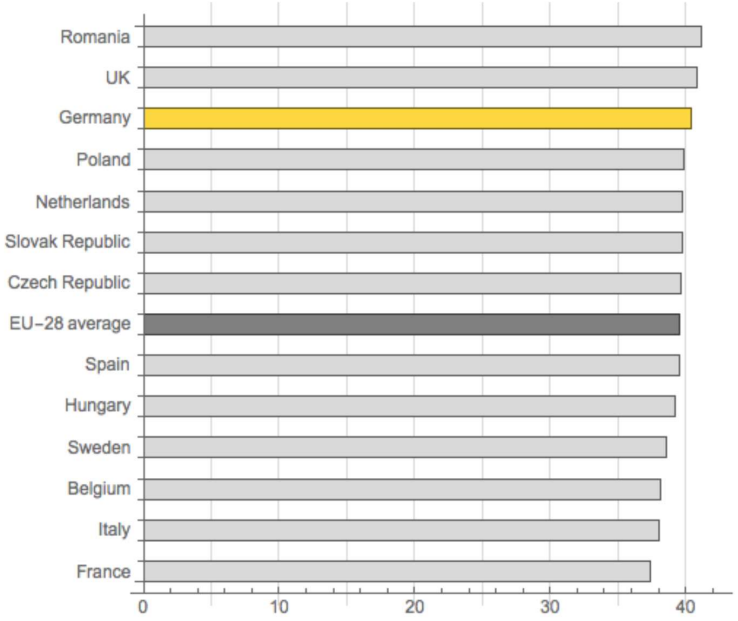
Data scoure: Eurofound 2014

https://github.com/ubco-mds-2022/Data-550

https://github.com/ubco-mds-2022/Data-550

Graph redrawn

https://github.com/ubco-mds-2022/Data-550



How about now?

The redrawn graph with an axis going all the way to zero. Source of images: *Misleading axes on graphs* callingbulls***.org

https://github.com/ubco-mds-2022/Data-550

What makes figures bad

Kieran Healy argues problems tend to come in three varieties[1]

1. *aesthetic* graphs are tacky, tasteless, or a hodgepodge of ugly or inconsistent design choices, unnecessary distractions
2. *substantive* the wrong data being presented
3. *perceptual* confusing or misleading because of how people perceive and process what they are looking at

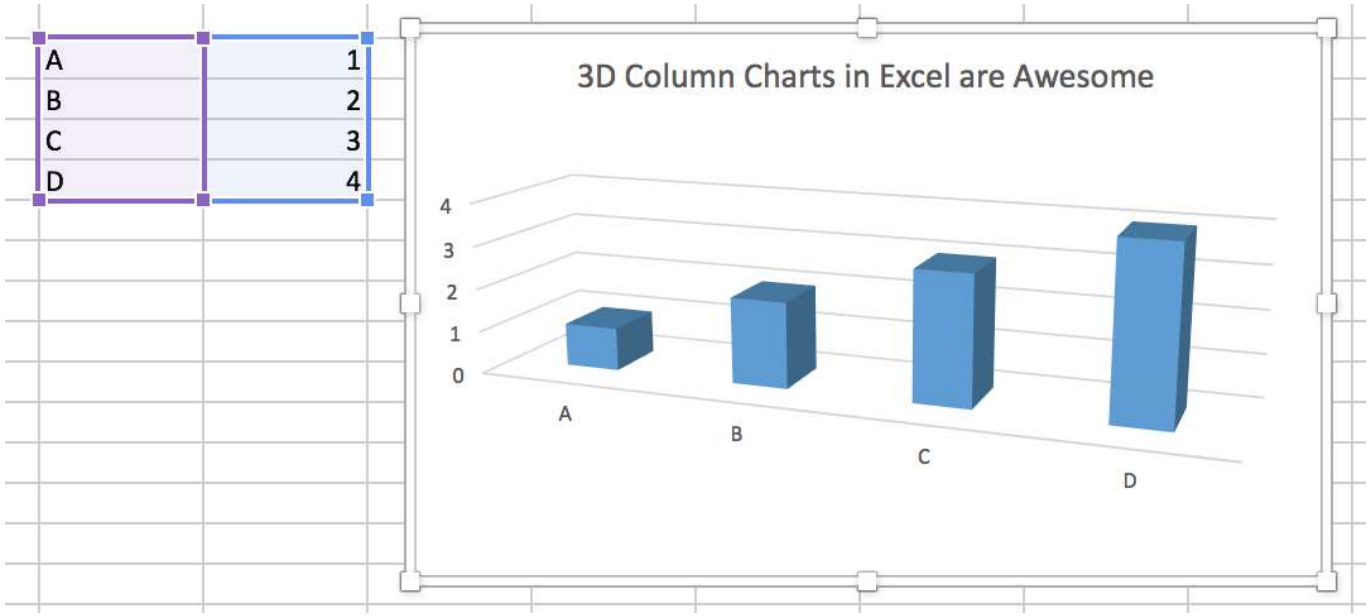
1. see Section 1 – 1.2 of Data Visualization: A practical introduction https://github.com/ubco-mds-2022/Data-550

Bad asethetic example

A chart with a considerable amount of junk in it. Source: Data Visualization A practical introduction by Kieran Healy

https://github.com/ubco-mds-2022/Data-550

Bad Perception example



A 3-D Column Chart created in Microsoft Excel for Mac. Although it may seem hard to believe, the values shown in the bars are 1, 2, 3, and 4. Source: Data Visualization A practical introduction by Kieran Healy

Useful junk?

One take on data viz priorities clarity over visual appeal.

https://github.com/ubco-mds-2022/Data-550

Embellished

'Monstrous Costs' by Nigel Holmes.

Plain version:

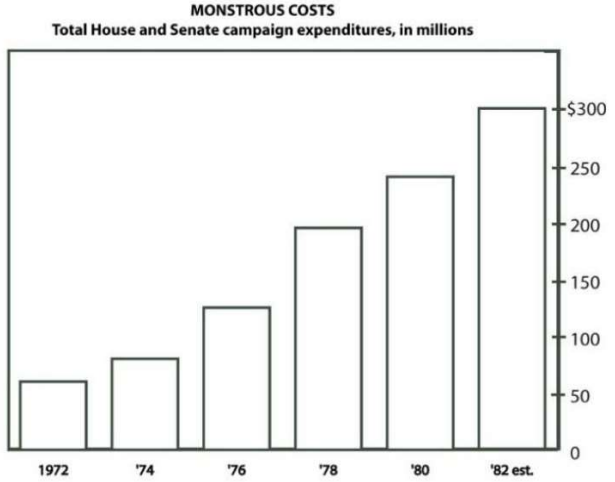


Image source: Useful Junk?

https://github.com/ubco-mds-2022/Data-550

KISS

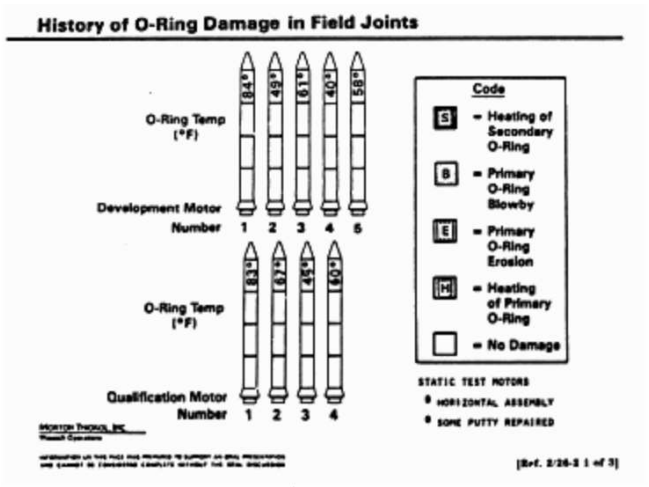
> “Graphical excellence is that which gives the viewer the greatests number of ideas in the shortest time with the least in in the smallest space”

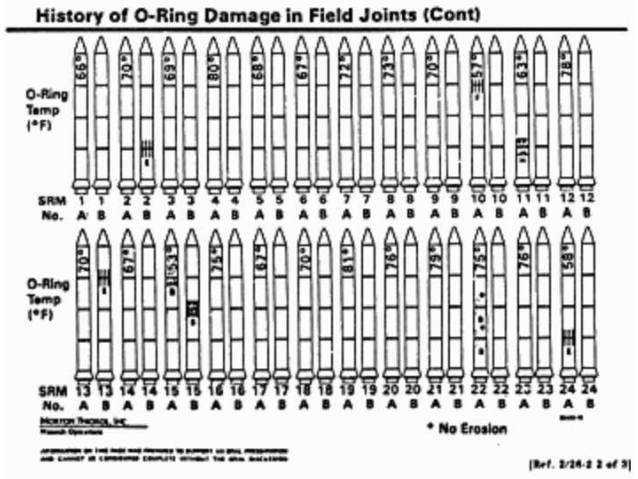
- Edward Tufte

https://github.com/ubco-mds-2022/Data-550

1986 Challenger disaster

A famous example involves the *Challenger* Diaster in which a space shuttle exploded as a result of damaged O-rings.





https://github.com/ubco-mds-2022/Data-550

Image source: Presidential Commission on the Space Shuttle Challenger Accident, vol. 5

https://github.com/ubco-mds-2022/Data-550

The O-ring data

mission <int>	temperature <int>	damaged <int>	undamaged <dbl>
1	53	5	1
2	57	1	5
3	58	1	5
4	63	1	5
5	66	0	6
6	67	0	6
7	67	0	6

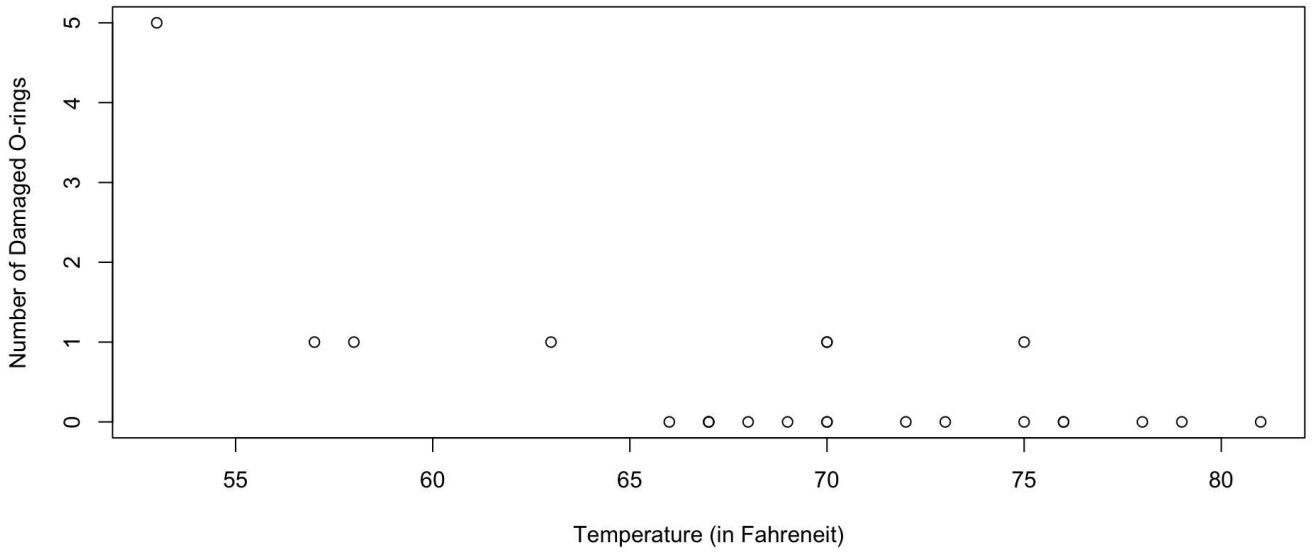
https://github.com/ubco-mds-2022/Data-550

mission <int>	temperature <int>	damaged <int>	undamaged <dbl>
8	67	0	6
9	68	0	6
10	69	0	6

1-10 of 23 rows Previous **1** 2 3 Next

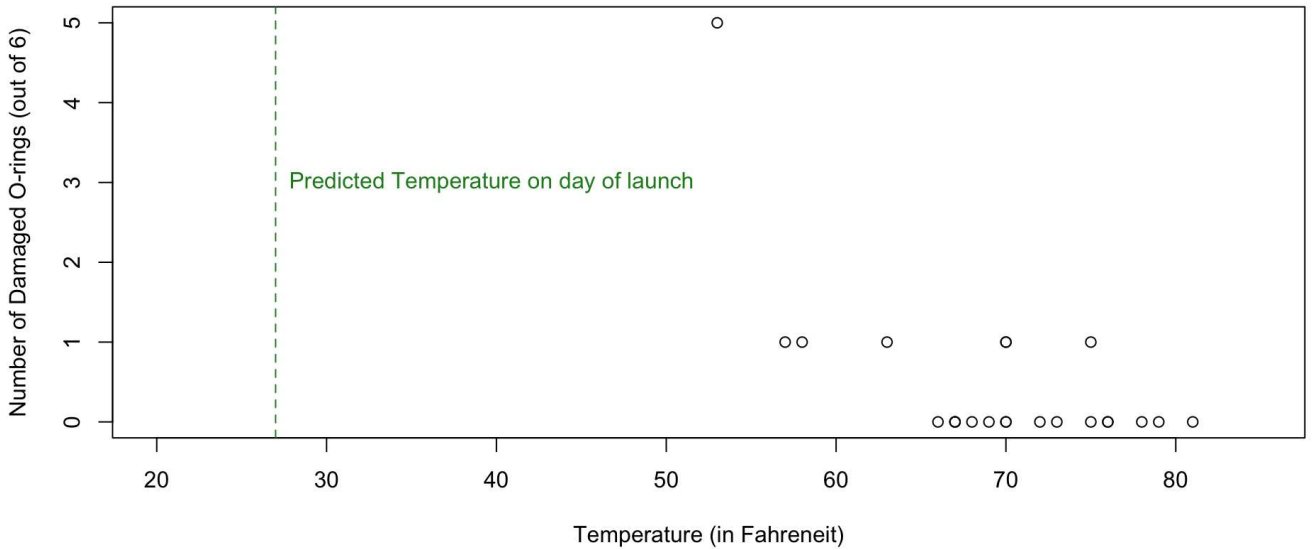
https://github.com/ubco-mds-2022/Data-550

Challenger Scatter plot



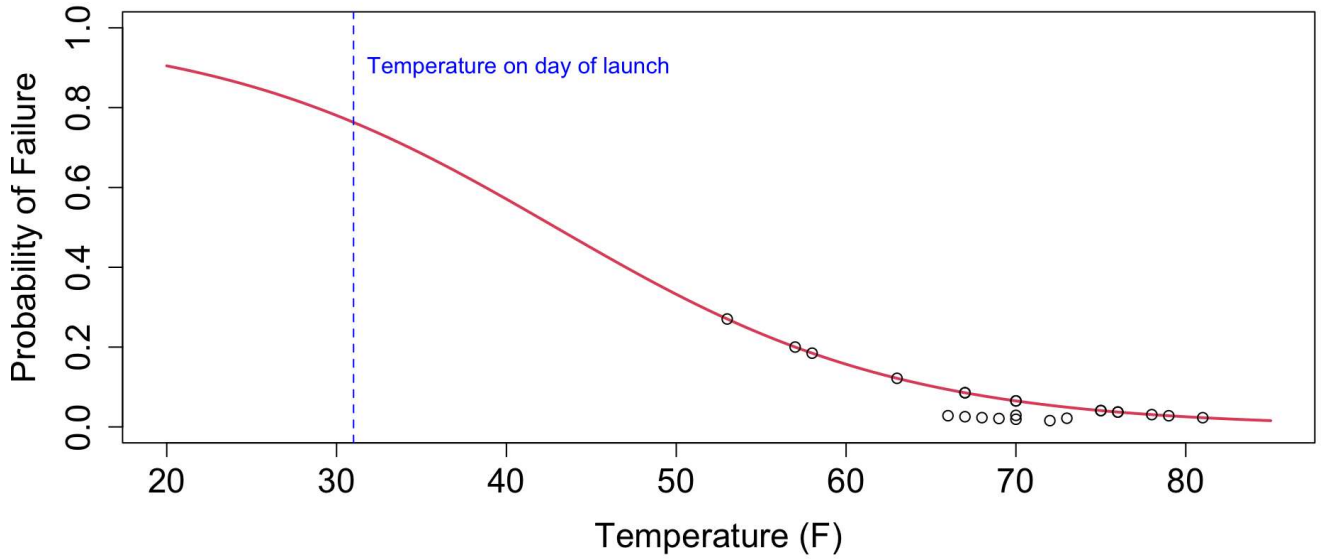
https://github.com/ubco-mds-2022/Data-550

Predicted Temperature



https://github.com/ubco-mds-2022/Data-550

Challenger Analysis



https://github.com/ubco-mds-2022/Data-550

Why have a human in the loop?

> "Computer-based visualization systems provide visual representations of datasets designed to help **people** carry out tasks more effectively". (Tamara Munzer)

Visualization is suitable when there is a need to *augment* human capabilities rather than replace people with computational decision-making methods.

https://github.com/ubco-mds-2022/Data-550

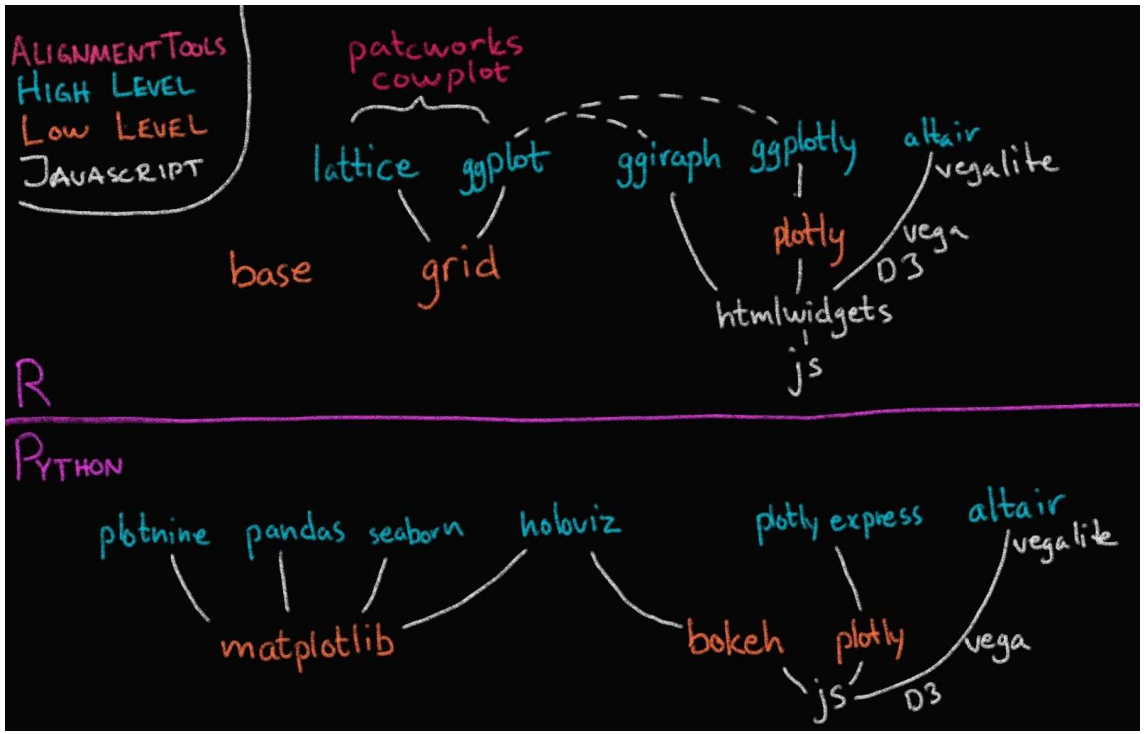
How to Visualize Data

There are two general types of visualization approaches

1. Imperative (low-level): focus on plot construction details
    - minute control over plotting details, but laborious for complex visualizations
2. Declarative (high-level): focuses on the data
    - often includes linking columns to visual *channels* and has smart defaults without complete control over minor plotting details

https://github.com/ubco-mds-2022/Data-550

Plotting in R and Python



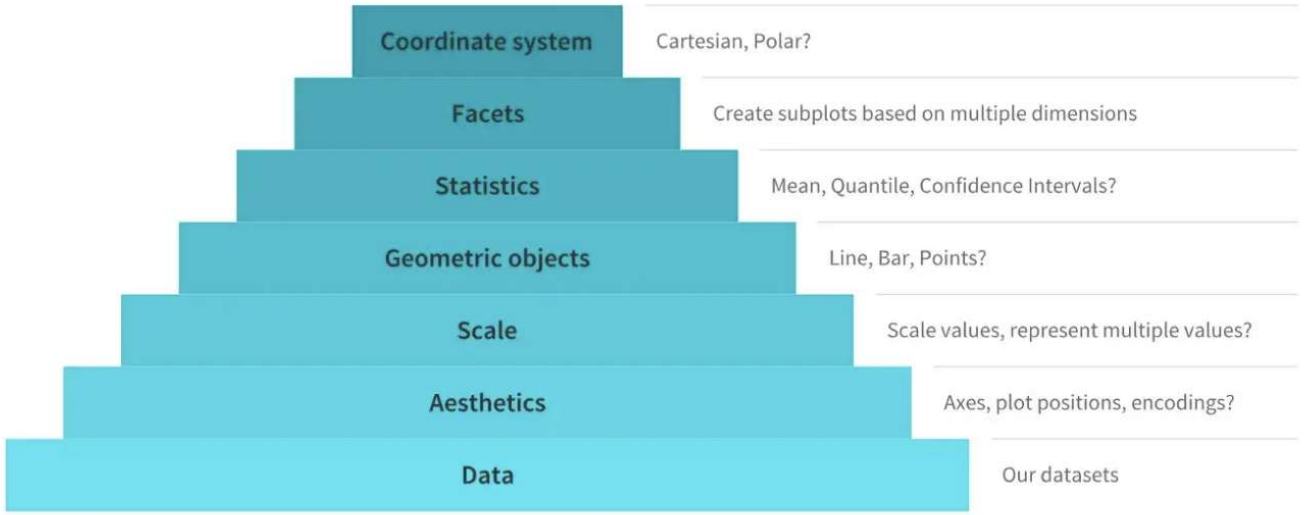
https://github.com/ubco-mds-2022/Data-550

Grammar of Graphics

- “Grammar” is a set of structural rules which help define and establish the components of a language
- A high-level *grammar of graphics* [1] is a framework which help define and establish the components of a graphic
- Simple grammatical components combine to create visualizations through the use of *channels*
- We describe and construct visualizations using a layered and structured approach

1. see Grammar of Graphics (2005) by Leland Wilkinson, Graham Wills

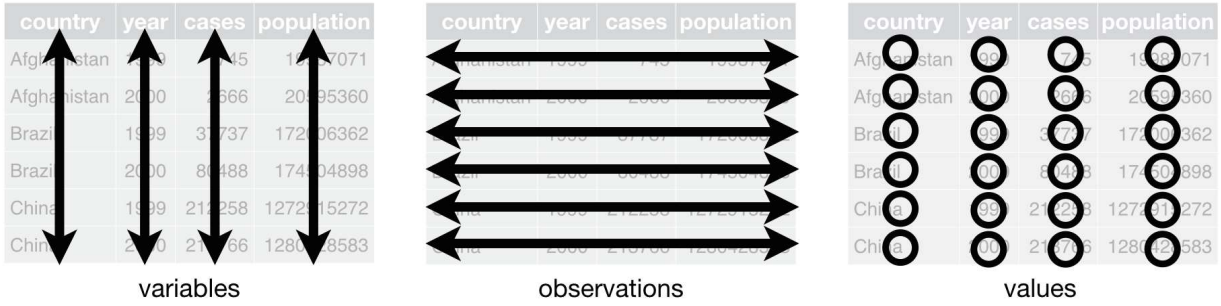
Components of the Grammar of Graphics



Major components of the Grammar of Graphics depicted using a pyramid architecture to show an inherent layered hierarchy of components. Image source: Dipanjan (DJ) Sarkar

1. Data

- Always start with the data
- We will be working with data frames in the *tidy* data format



Rules of a dataset tidy: variables are in columns, observations are in rows, and values are in cells. Image Source R for Data Science by Hadley Whickham

https://github.com/ubco-mds-2022/Data-550

2. Aesthetics

- Select the axes based on the data dimensions you want to visualize
- Select positions of various data points in the plot
- Check if any form of encoding is needed
  - eg. size, shape, color and so on
  - these are particularly useful for plotting multiple data dimensions

https://github.com/ubco-mds-2022/Data-550

3. Scale

Decide if you need to:

- scale the potential values
- use a specific scale to represent multiple values or a range of values

https://github.com/ubco-mds-2022/Data-550

4. Geometric objects

- These are popularly known as `geoms`
- This would cover how data points are depicted on the visualization, eg. points, bars, lines and so on

https://github.com/ubco-mds-2022/Data-550

5. Statistics

- Do we need to show some statistical measures in the visualization such as measures of central tendency (mean/median), spread (eg. variance), confidence intervals, ...

6. Facets

- Do we need to create subplots based on specific data dimensions?

7. Coordinate system

- What kind of a coordinate system should the visualization be based on?
- should it be Cartesian or polar?

https://github.com/ubco-mds-2022/Data-550

Grammar of Graphics in R/Python

- Two of the most prominent declarative statistical visualization libraries are:
  1. `altair` (Python and R) and
  2. `ggplot2`[1] (in R) / `plotnine`[2] (in Python)
- These offer a powerful and concise visualization grammar for quickly building a wide range of statistical graphics.

1. **A Layered grammar of Graphics** by Hadley Wickham *Journal of Computational and Graphical Statistics*, vol. 19, no. 1, pp. 3–28, 2010
2. There is also **`ggplot`** library for Python, however, there are backward incompatibility issues with newer versions of **`pandas`**

Create a canvas/chart

MPG (Q)	Origin (N)	Rank (N)	Year (T)
27.0	USA	1	1982-01-01
44.0	Japan	10	1972-01-01
32.0	Japan	221	1970-01-01
28.0	Europe	4	2023-01-01
31.0	USA	53	1980-01-01

```
ggplot(data)
alt.Chart(data)
```

https://github.com/ubco-mds-2022/Data-550

Create a canvas/chart

Encode visual aesthetic

MPG (Q)	Origin (N)	Rank (N)	Year (T)
27.0	USA	1	1982-01-01
44.0	Japan	10	1972-01-01
32.0	Japan	221	1970-01-01
28.0	Europe	4	2023-01-01
31.0	USA	53	1980-01-01

- x-axis
- y-axis
- colour
- size
- shape ...

```
ggplot(data,aes(x,y))
alt.Chart(data)          .encode(x,y)
```

https://github.com/ubco-mds-2022/Data-550

Create a canvas/chart

Encode visual aesthetic

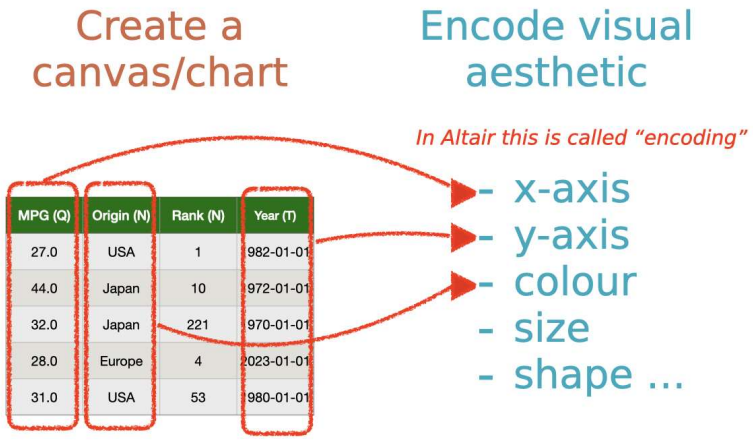
In ggplot2 this is called "mapping"

MPG (Q)	Origin (N)	Rank (N)	Year (T)
27.0	USA	1	982-01-01
44.0	Japan	10	972-01-01
32.0	Japan	221	970-01-01
28.0	Europe	4	2023-01-01
31.0	USA	53	980-01-01

- x-axis
- y-axis
- colour
- size
- shape ...

```
ggplot(data,mapping=aes(x,y))
```

https://github.com/ubco-mds-2022/Data-550




```
ggplot(data,mapping=aes(x,y))
alt.Chart(data)          .encode(x,y)
```

See Altair's encoding channel options here

https://github.com/ubco-mds-2022/Data-550

Create a canvas/chart

Encode visual aesthetic

Add geometric marks

MPG (Q)	Origin (N)	Rank (N)	Year (T)
27.0	USA	1	1982-01-01
44.0	Japan	10	1972-01-01
32.0	Japan	221	1970-01-01
28.0	Europe	4	2023-01-01
31.0	USA	53	1980-01-01

- x-axis
- y-axis
- colour
- size
- shape ...

* points
* lines
* Box-plot
* bar

```
ggplot(data,aes(x,y)) + geom_points()
alt.Chart(data).mark_points().encode(x,y)
```

https://github.com/ubco-mds-2022/Data-550

Create a canvas/chart

MPG (Q)	Origin (N)	Rank (N)	Year (T)
27.0	USA	1	1982-01-01
44.0	Japan	10	1972-01-01
32.0	Japan	221	1970-01-01
28.0	Europe	4	2023-01-01
31.0	USA	53	1980-01-01

Encode visual aesthetic

- x-axis
- y-axis
- colour
- size
- shape ...

Add geometric marks

* points
* lines
* Box-plot
* bar

```
ggplot(data,aes(x,y)) + geom_*()
alt.Chart(data) .mark_*().encode(x,y)
```

see ggplot geom_ functions options here and Altair graphical mark_ options here

https://github.com/ubco-mds-2022/Data-550

ggplot

- We'll do a quick demo in `ggplot()` from the *ggplot2* library and then cover the same data using Altair in Python (separate slide deck)
- A very helpful resource is the ggplot2 cheat sheet
- It can be a little bit overwhelming at first, but by working through these simple examples, I hope to give you a feel for it.

https://github.com/ubco-mds-2022/Data-550

Cars data

Model <chr>	MPG <dbl>	Cyli
chevrolet chevelle malibu	18.0	
buick skylark 320	15.0	
plymouth satellite	18.0	
amc rebel sst	16.0	
ford torino	17.0	
ford galaxie 500	15.0	
chevrolet impala	14.0	

https://github.com/ubco-mds-2022/Data-550

Model <chr>	MPG <dbl>	Cyli
plymouth fury iii	14.0	
pontiac catalina	14.0	
amc ambassador dpl	15.0	

1-10 of 406 rows | 1-4 of 9 columns Previous **1** 2 3 4 5 6 … 41 Next

https://github.com/ubco-mds-2022/Data-550

Create a canvas/chart

MPG (Q)	Origin (N)	Rank (N)	Year (T)
27.0	USA	1	1982-01-01
44.0	Japan	10	1972-01-01
32.0	Japan	221	1970-01-01
28.0	Europe	4	2023-01-01
31.0	USA	53	1980-01-01

```
ggplot(data)
alt.Chart(data)
```

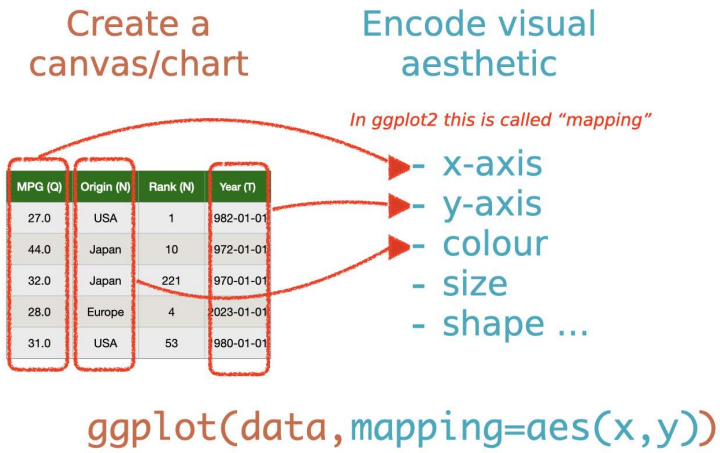
https://github.com/ubco-mds-2022/Data-550

canvas code

```
1 ggplot(cars)
```


https://github.com/ubco-mds-2022/Data-550

Map columns to visual asethetics (encode)

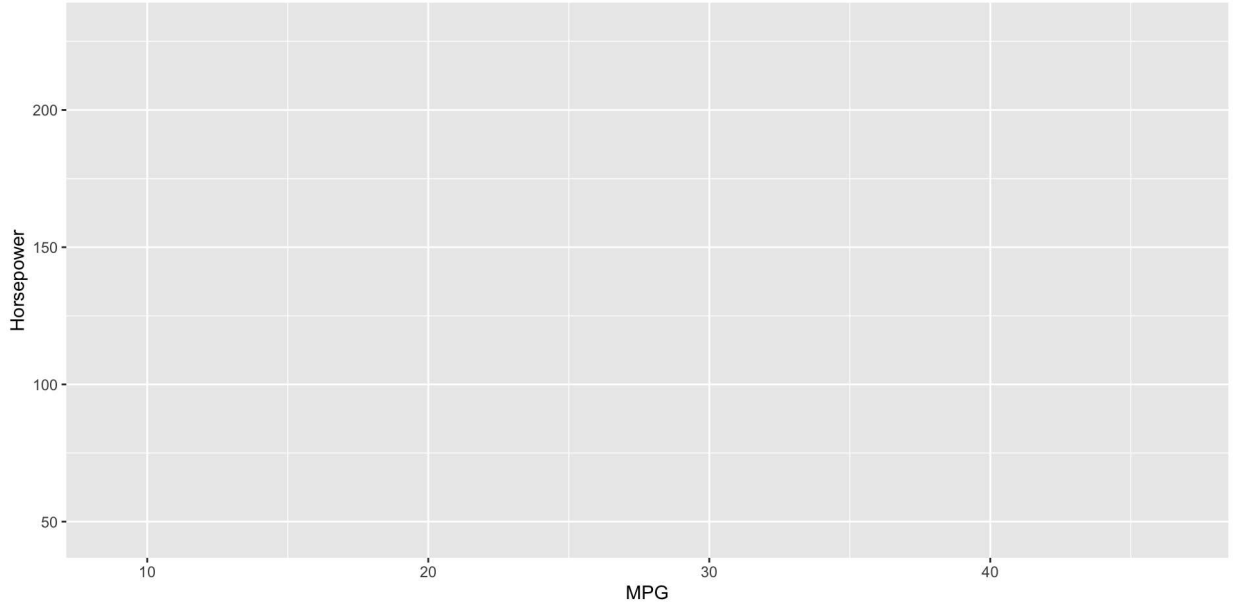


https://github.com/ubco-mds-2022/Data-550

mapping code

```
ggplot(cars, mapping = aes(x = MPG, y = Horsepower))
```

```
# same as ggplot(cars, aes(x = MPG))
```



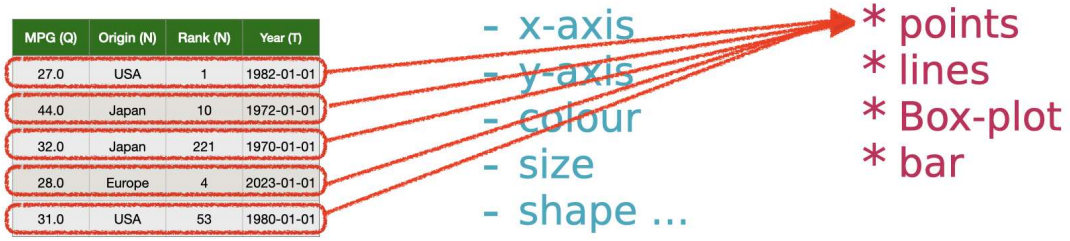
https://github.com/ubco-mds-2022/Data-550

Add geometric markings

Create a canvas/chart

Encode visual aesthetic

Add geometric marks

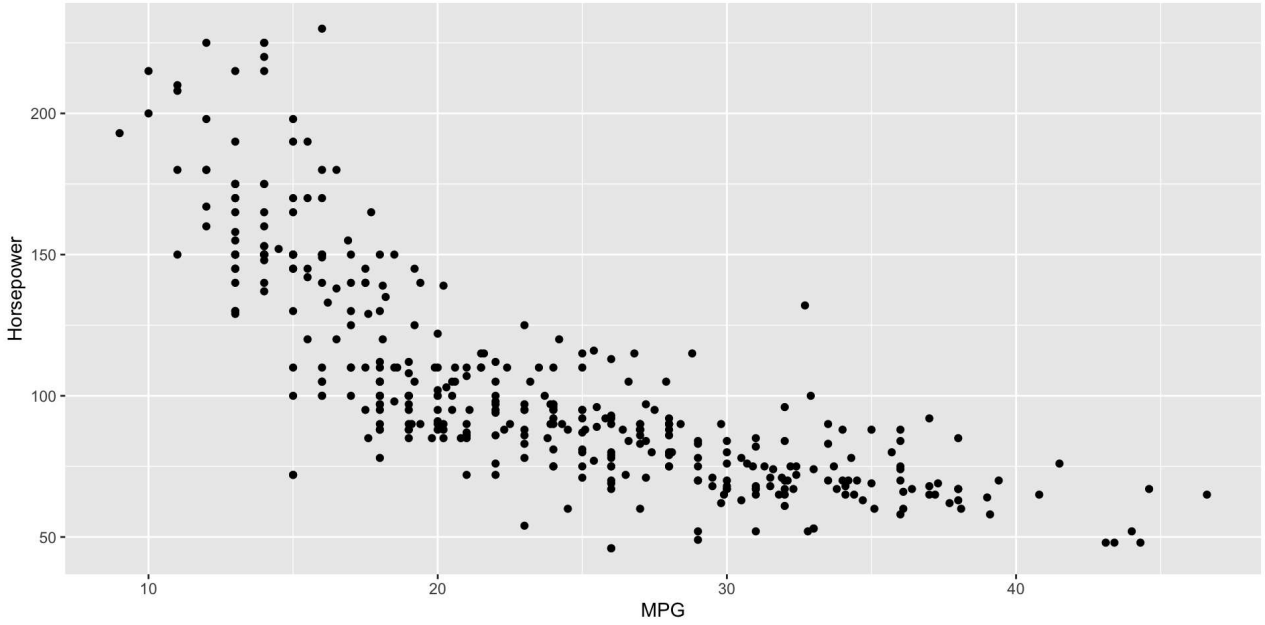



```
ggplot(data,aes(x,y)) + geom_points()
alt.Chart(data).mark_points().encode(x,y)
```

https://github.com/ubco-mds-2022/Data-550

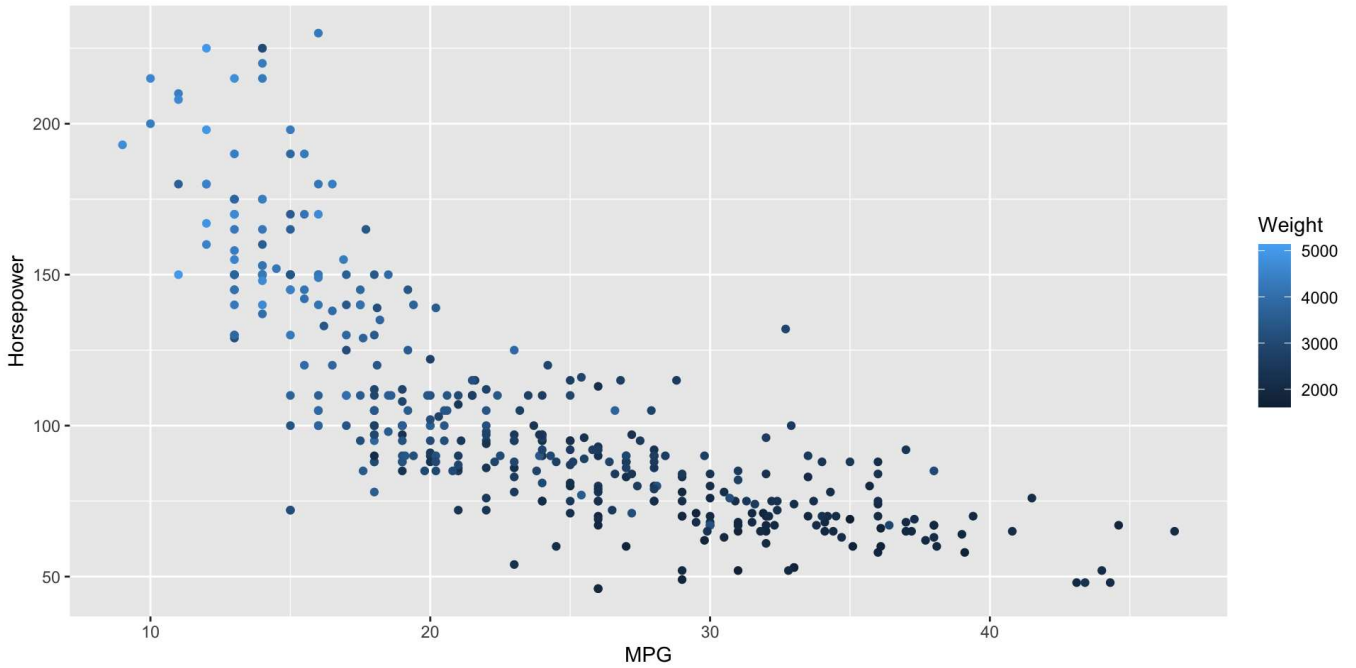
Scatterplot

```
ggplot(cars, aes(x = MPG, y = Horsepower)) +
    geom_point()
```



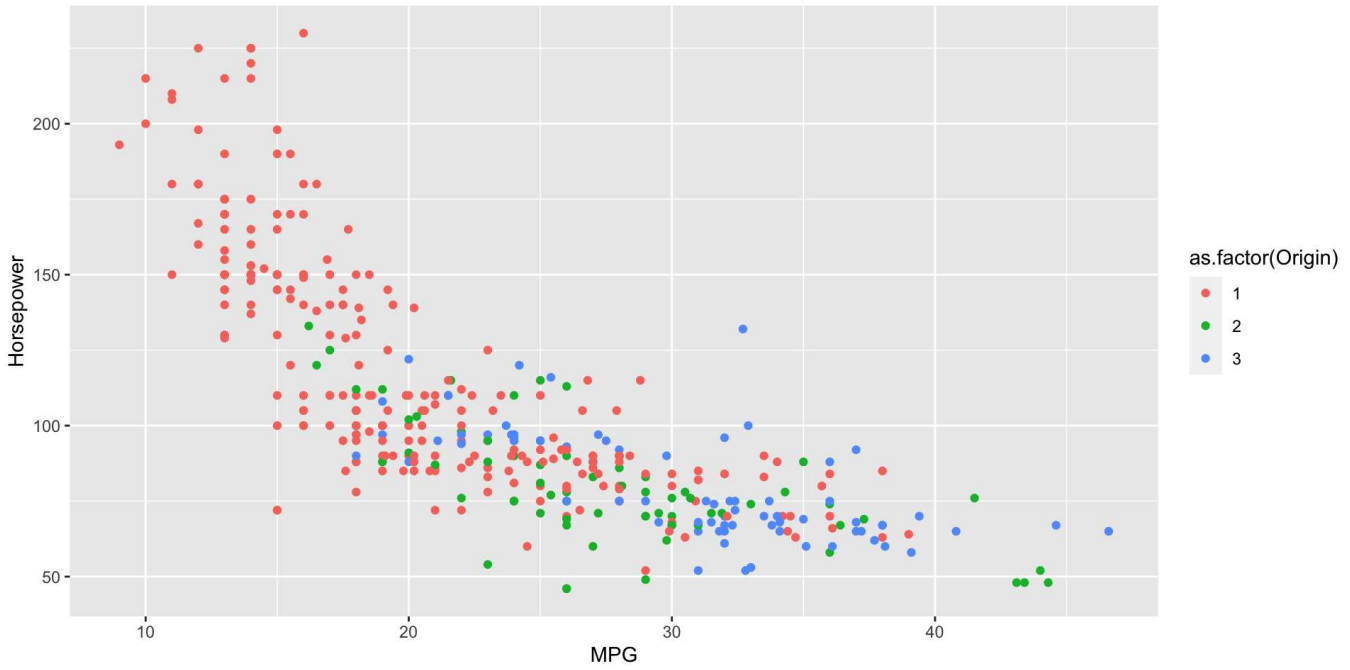
https://github.com/ubco-mds-2022/Data-550

```
ggplot(cars, aes(x = MPG, y = Horsepower, color = Weight)) +
    geom_point()
```



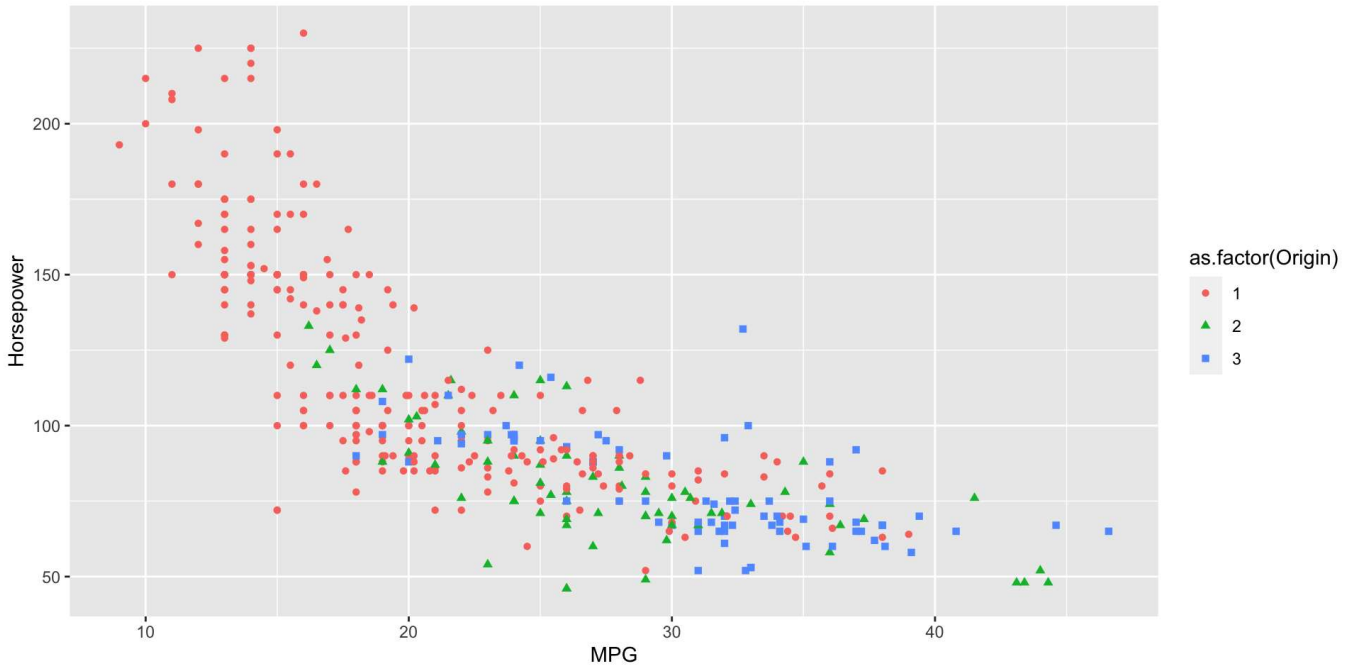
https://github.com/ubco-mds-2022/Data-550

```
ggplot(cars, aes(x = MPG, y = Horsepower, color = as.factor(Origin))) +
    geom_point()
```



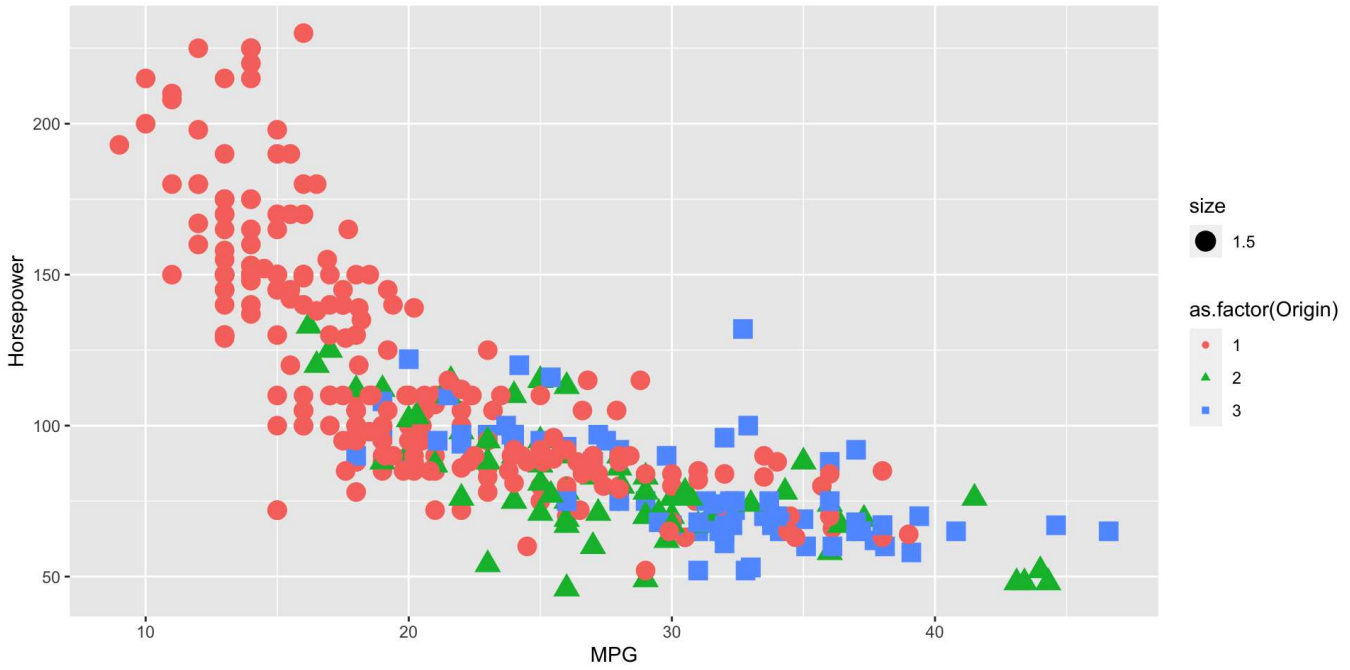
https://github.com/ubco-mds-2022/Data-550

```
ggplot(cars, aes(x = MPG, y = Horsepower, color = as.factor(Origin))) +
    geom_point(aes(shape = as.factor(Origin)))
```



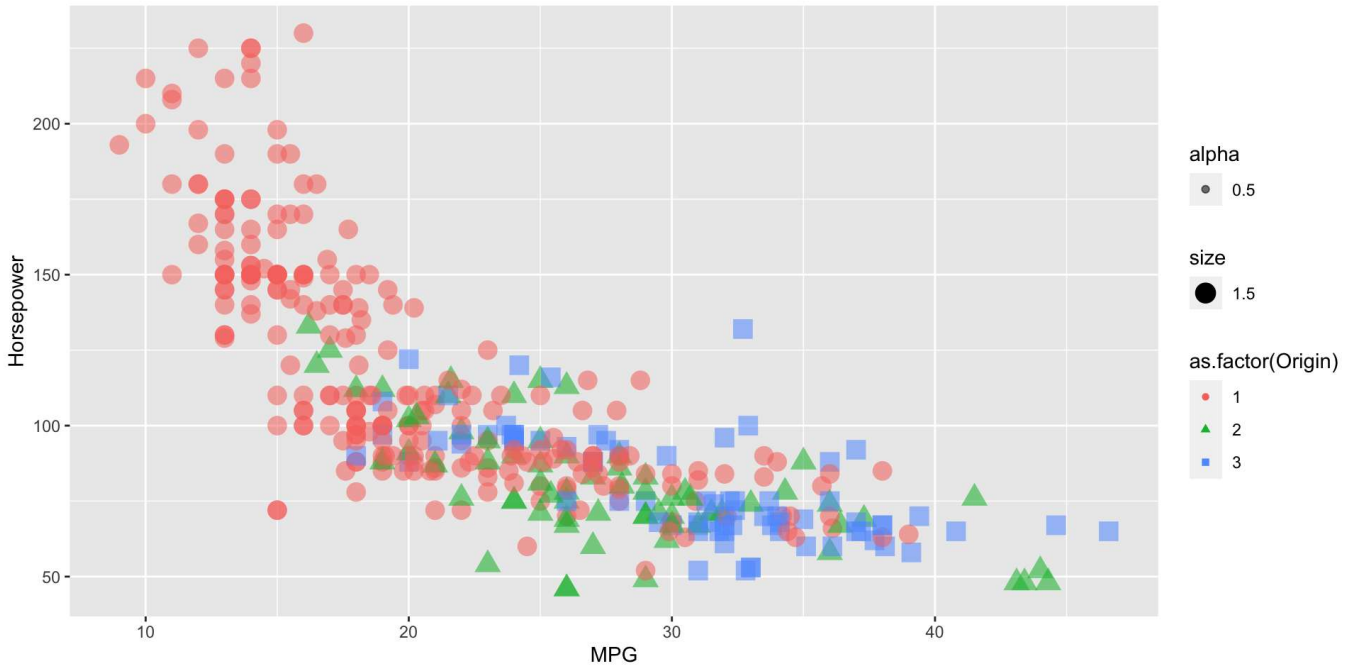
https://github.com/ubco-mds-2022/Data-550

```
ggplot(cars, aes(x = MPG, y = Horsepower, color = as.factor(Origin))) +
    geom_point(aes(shape = as.factor(Origin), size = 1.5))
```



https://github.com/ubco-mds-2022/Data-550

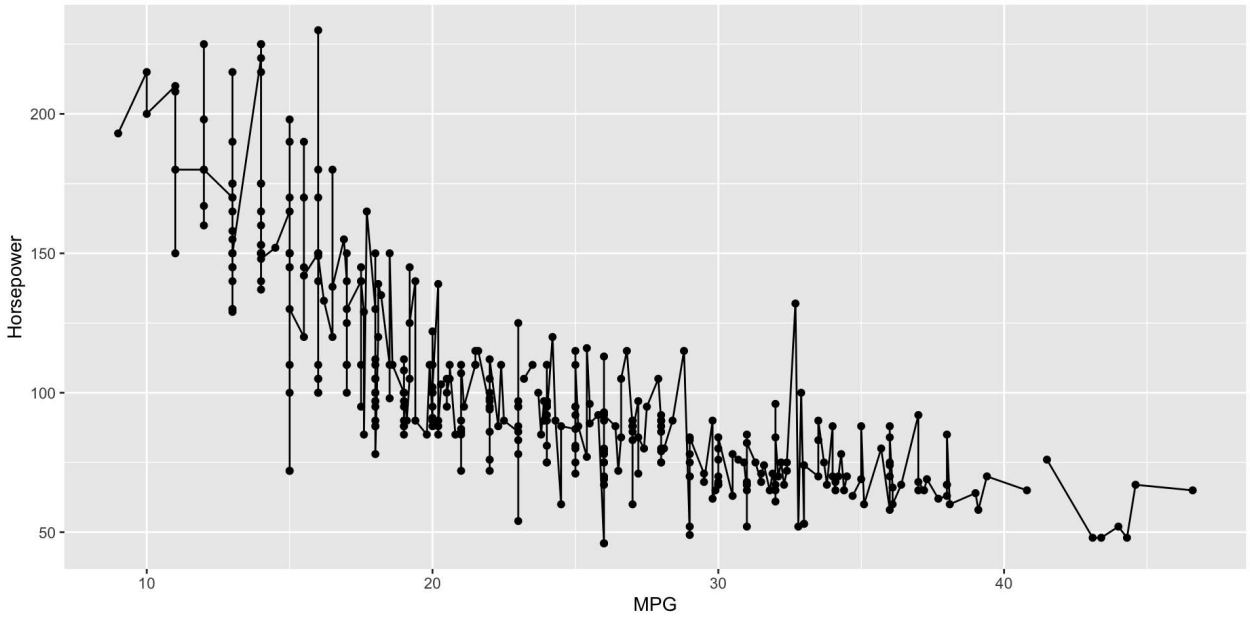
```
ggplot(cars, aes(x = MPG, y = Horsepower, color = as.factor(Origin))) +
    geom_point(aes(shape = as.factor(Origin), size = 1.5, alpha = 0.5))
```



https://github.com/ubco-mds-2022/Data-550

```r
base_plot = ggplot(cars, aes(x = MPG, y = Horsepower))

base_plot +
  geom_point() +
  geom_line()
```



https://github.com/ubco-mds-2022/Data-550

https://github.com/ubco-mds-2022/Data-550