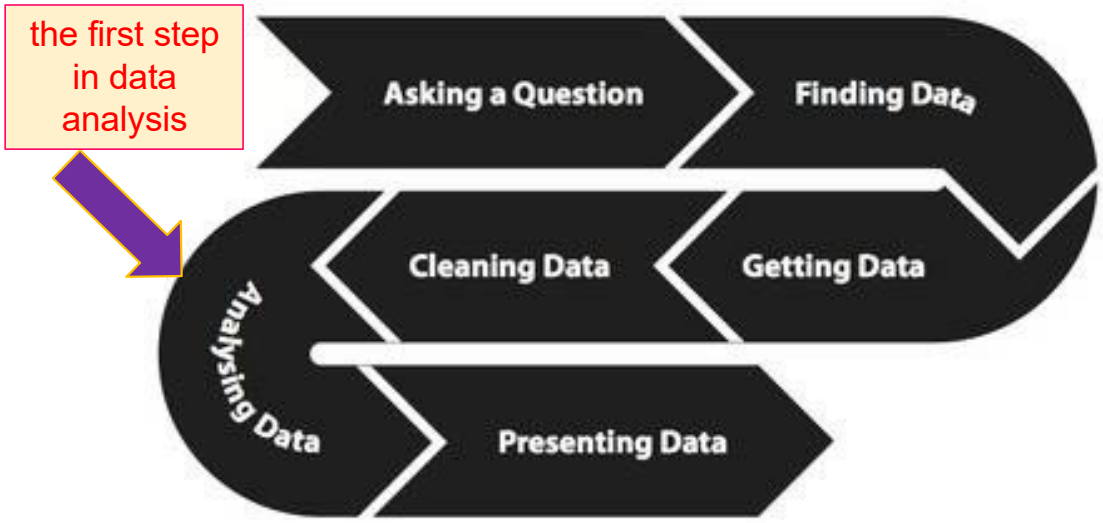
Exploratory Data Analysis

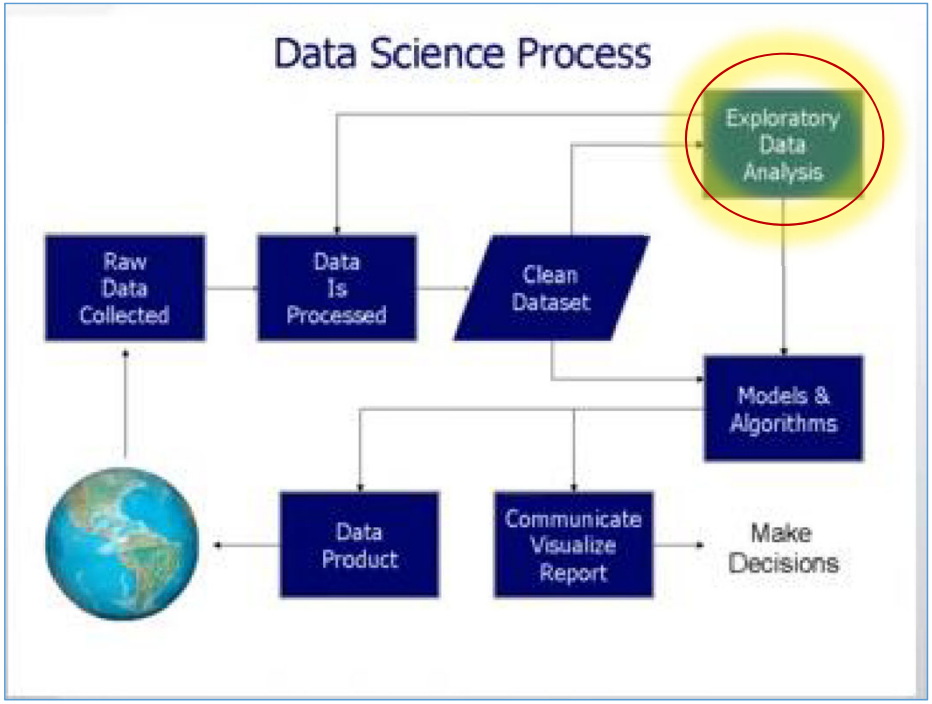
WQD7001

Learning Objectives

1) To explain the concept of exploratory data analysis (EDA).
2) To describe the roles of EDA.
3) To discuss the 6 principles of analytical design.
4) To differentiate several types of data visualizations.
5) To explore types of graphs and plotting systems in R.
6) To review graphics devices.

Exploratory Data Analysis (EDA)





Why we Analyze Data?

i. To understand what has happened or what is happening.
ii. To predict what is likely to happen, in future or in other circumstances we haven't seen.
iii. To guide us in making decision.

Type of Analysis Results

Informative - For example plots, or any long variable summary.

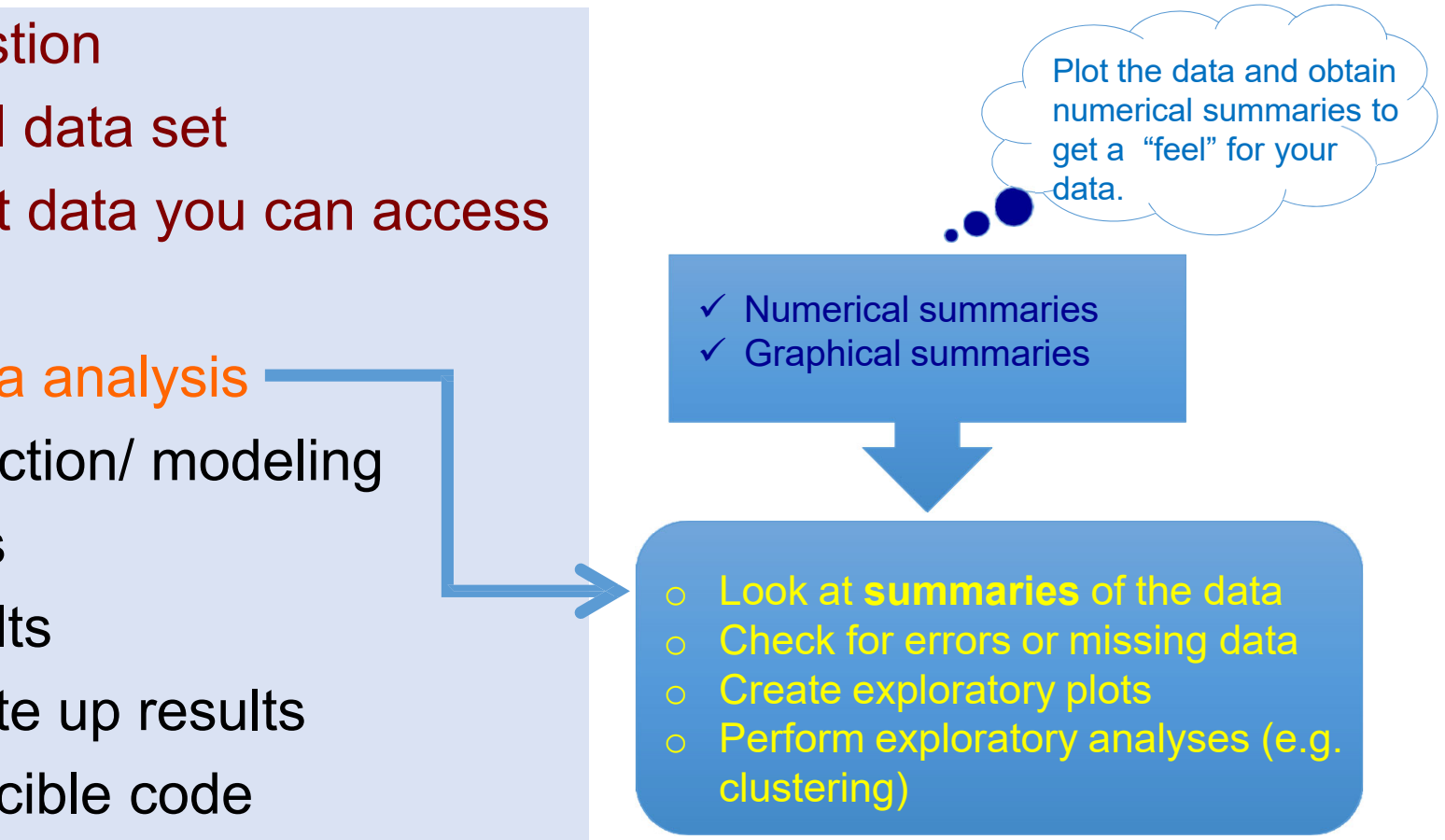
- We cannot filter data from it, but give us a lot of information at once.
- Most used on the **EDA stage**.

Operative - The results can be used to take an action directly on the data workflow

- e.g., selecting any variables whose percentage of missing values are below 20%.
- Most used in the **Data Preparation stage**.

Steps in a Data Processing / Analysis

- ✧Define the question
- ✧Define the ideal data set
- ✧Determine what data you can access
- ✧Obtain the data
- ✧Exploratory data analysis
- ✧Statistical prediction/ modeling
- ✧Interpret results
- ✧Challenge results
- ✧Synthesize/ write up results
- ✧Create reproducible code



pm25	fips	region	longitude	latitude
9.771185	1003	east	-87.7483	30.59278
9.993817	1027	east	-85.8429	33.26581
10.68862	1033	east	-87.726	34.73148
11.33742	1049	east	-85.7989	34.45913
12.11976	1055	east	-86.0321	34.0186
10.8278	1069	east	-85.3504	31.18973
11.58393	1073	east	-86.8281	33.52787
11.262	1089	east	-86.5882	34.73079
9.414423	1097	east	-88.1397	30.72226
11.39149	1103	east	-86.9189	34.50702
12.38479	1113	east	-85.1011	32.376
10.6495	1117	east	-86.6987	33.26912
11.33382	1121	east	-86.1783	33.3685
12.30244	1125	east	-87.5117	33.2356
11.02451	1127	east	-87.2854	33.81989
6.05886	2020	west	-149.762	61.1919
11.10147	2090	west	-147.568	64.81859
7.308113	2110	west	-134.512	58.35142
7.147626	2170	west	-149.481	61.76274
6.929844	4003	west	-109.904	31.75027
6.132351	4005	west	-111.511	35.77144
8.228339	4013	west	-112.088	33.49451

Given this dataset, what would you like to know about it?

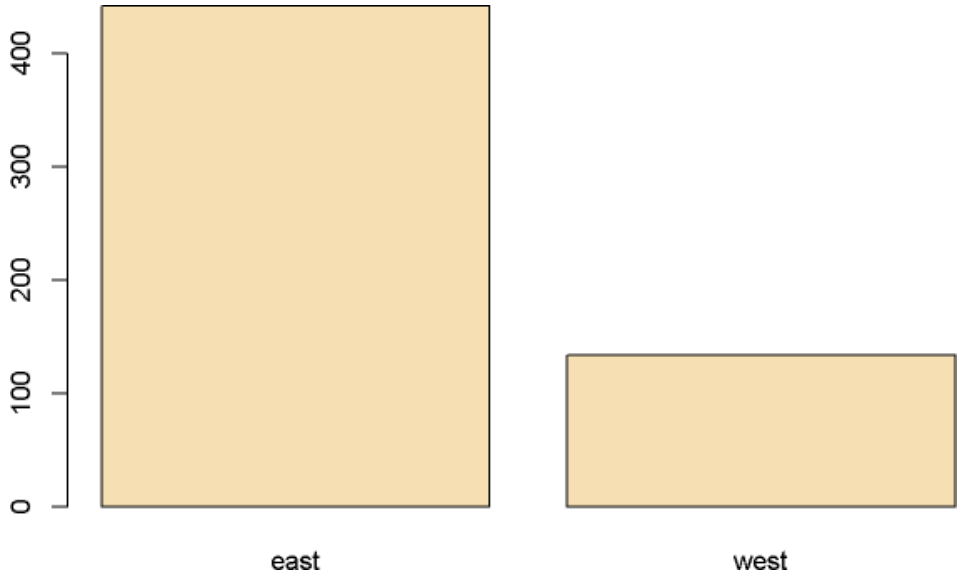
Five Number Summary

```
summary(pollution$pm25)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   3.383   8.549  10.050   9.836  11.360  18.440
```

We can see that the median across all the counties in the dataset is about 10 micrograms per cubic meter.

Bar Plot



We can see quite clearly that there are many more counties in the eastern U.S. in this dataset than in the western U.S.

Discover more from here: https://rpubs.com/profversaggi/eda-exploratory-graphs

According to Techopedia:

"Exploratory data analysis (EDA) is a term for certain kinds of initial analysis and findings done with data sets, usually early on in an analytical process. Some experts describe it as "taking a peek" at the data to understand more about what it represents and how to apply it.

Exploratory data analysis is often a precursor to other kinds of work with statistics and data."

A First Look at the Data

EDA takes a **broad look** at data and tries to **make sense** of it.

EDA involves the analyst trying to **get a "feel" for the data set**, often using their own judgment to determine what the most important elements in the data set are.

EDA is used to **understand and summarize** the contents of a dataset, usually to investigate a specific question or to prepare for more advanced modeling.

EDA typically relies heavily on **visualizing the data** to **assess patterns** and **identify data characteristics** that the analyst would not otherwise know to look for.

It also takes advantage of a number of **quantitative methods to describe the data**.

The greatest value of a picture is when it forces us to notice what we never expected to see.
—John W. Tukey, on Exploratory Data Analysis

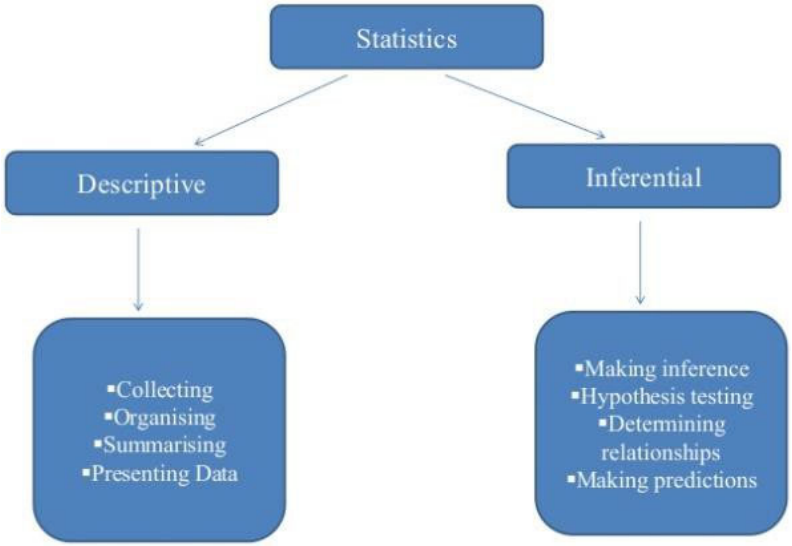
Exploratory data analysis, or EDA for short, is a term coined by John W. Tukey for describing the act of looking at data to see what it seems to say.

Confirmatory vs. Exploratory Data Analysis

Confirmatory data analysis

- ✓ tests a hypothesis (true or false)
- ✓ settles questions

(Inferential statistics)



Exploratory data analysis

- ✓finds a good description (new features discovered)
- ✓ raises new questions

(Descriptive statistics)

- ✓ Describe data
- ✓ Provides a summary
- ✓ aka: Summary statistics

Movie Runtime	
Statistic	**Value (minutes)**
Minimum	38
1st Quartile	93
Median	101
Mean	104
3rd Quartile	113
Maximum	219

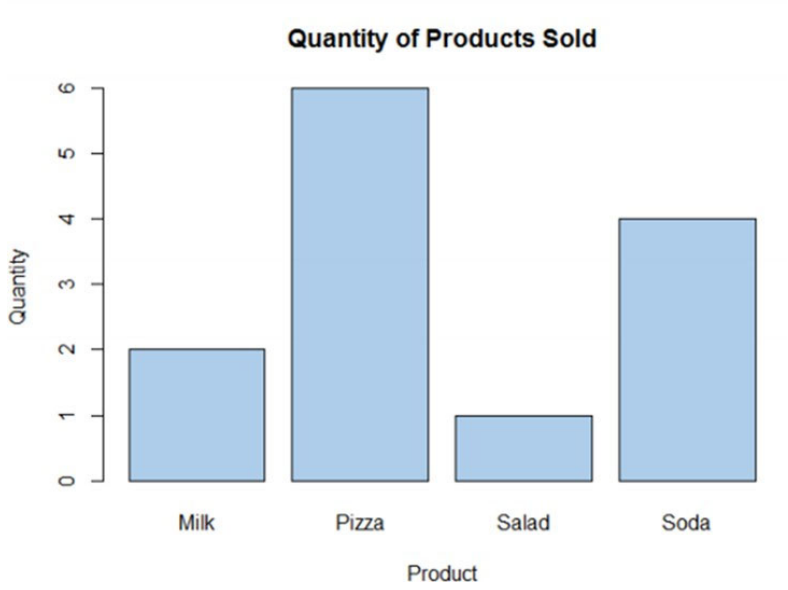
The Role of EDA

Explore a data set.

Use **methods** that help you **understand the data.**

- to help you **understand** the events that generated the data.
- to help you **see** what happened.

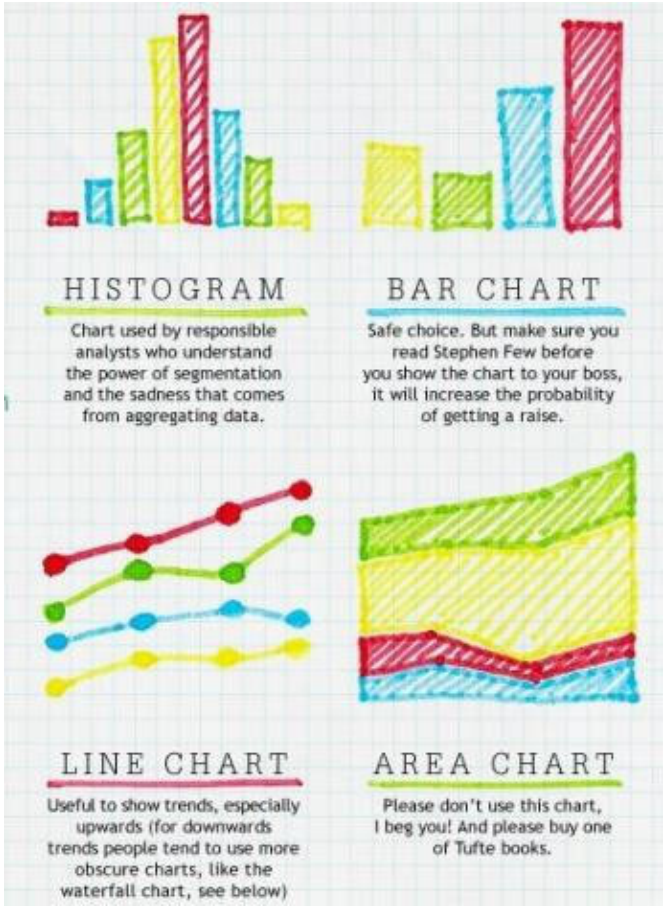
visualize

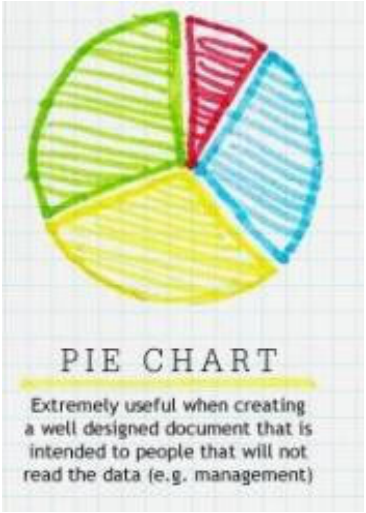


ID	Date	Customer	Product	Quantity
1	2015-08-27	John	Pizza	2
2	2015-08-27	John	Soda	2
3	2015-08-27	Jill	Salad	1
4	2015-08-27	Jill	Milk	1
5	2015-08-28	Miko	Pizza	3
6	2015-08-28	Miko	Soda	2
7	2015-08-28	Sam	Pizza	1
8	2015-08-28	Sam	Milk	1

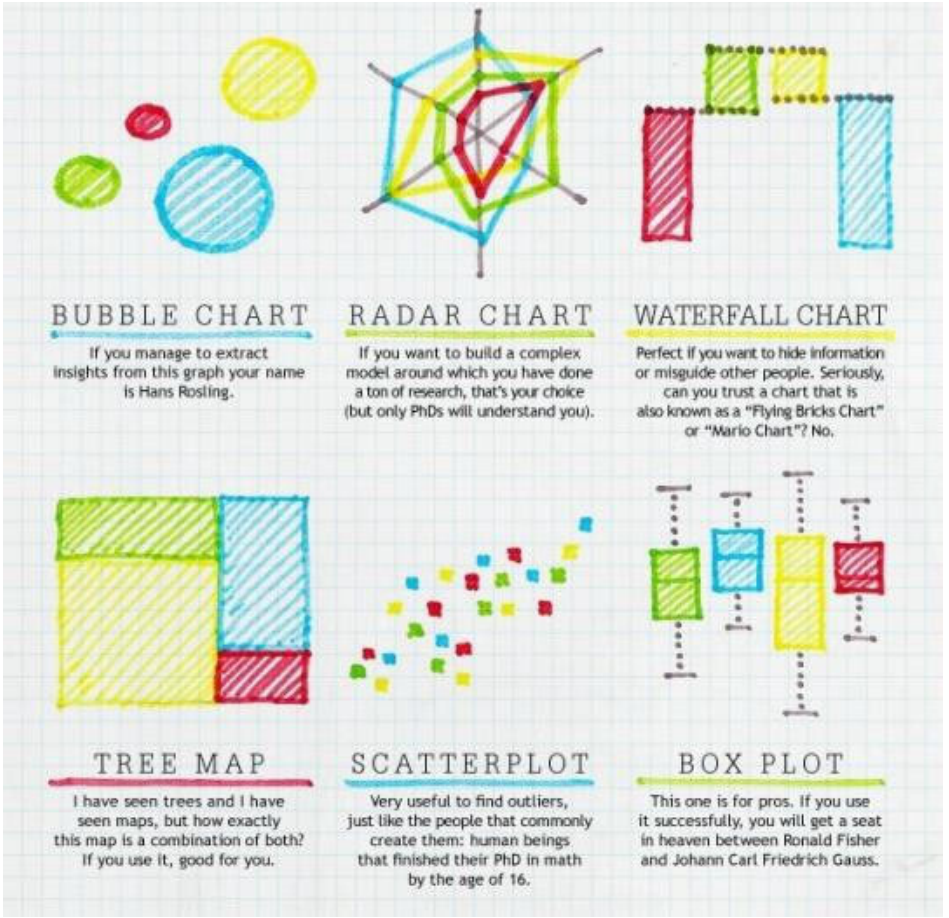
A (Good) Picture Is Worth a 1,000 Words

Visual Data Representation





FOR
Human pattern recognition



Purpose of Exploratory Data Analysis

Use **summary statistics** and **visualizations** to better understand data, and find clues about the tendencies of the data, its quality and to formulate assumptions and the hypothesis of our analysis.

EDA is **NOT** about making fancy visualizations or even aesthetically pleasing ones.

The **goal** is to try and **answer questions** with data.

Objectives of EDA

Critical process of performing initial investigations with the help of summary statistics and graphical representations on data so as to:

i. Determine the quality of data,
ii. Determine what statistical models can fit the data,
iii. Discover patterns,
iv. Detect outliers and anomalies,
v. Determine whether to apply univariate or multivariate analytical techniques.
vi. Check out if the assumptions about the data, that you or your team started out with is correct or way off.

EDA Checklist

- ❑ What question(s) are you trying to solve (or prove wrong)?
- ❑ What kind of data do you have and how do you treat different types?
- ❑ What's missing from the data and how do you deal with it?
- ❑ Where are the outliers and why should you care about them?
- ❑ How can you add, change or remove features to get more out of your data?

Reference: https://towardsdatascience.com/a-gentle-introduction-to-exploratory-data-analysis-f11d843b8184

Summary Statistics

- Measurements to **describe data**.
- In the field of **descriptive statistics**, there are many summary measurements.
- Examples of summary statistics for a **single numerical variable** is the *mean, median, mode, max, min, range, quartiles/percentiles, variance, standard deviation, coefficient of determination, skewness* and *kurtosis*.
- Summary statistics for **categorical variables** is the number of distinct *counts*.
- Most basic summary statistic for **text data** is term *frequency* and *inverse document frequency*.
- For **bivariate data**, the summary statistics is the *linear correlation, chi-squared*, or the *p value* based *z-test, t-test or analysis of variance*.

6 Principles of Analytical Design

Edward Tufte declares that "**The purpose of an evidence presentation is to assist thinking**", and that these **six principles** of analytical design "are derived from the principles of analytical thinking." (Beautiful Experience, p. 137).

Principle 1: Show comparisons, contrasts, differences

- always comparative (compared to what)
- randomized trial - compare control group to test group
- evidence for a hypothesis is always relative to another competing hypothesis

Principle 2: Show causality/mechanism/explanation/systematic structure

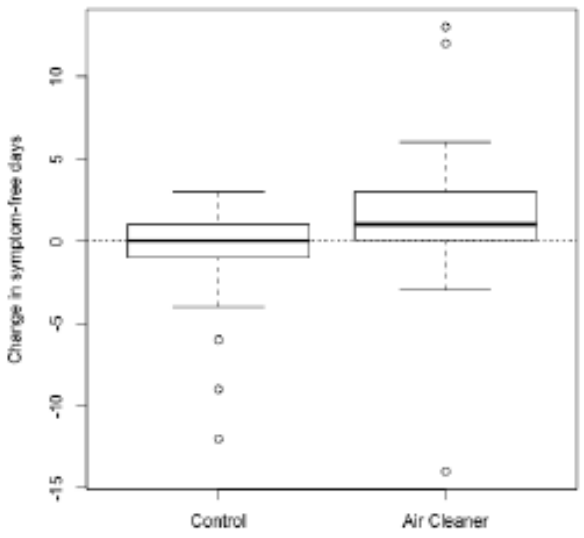
- form hypothesis to evidence showing a relationship (causal framework, why something happened)

Principle 3: Show multivariate data

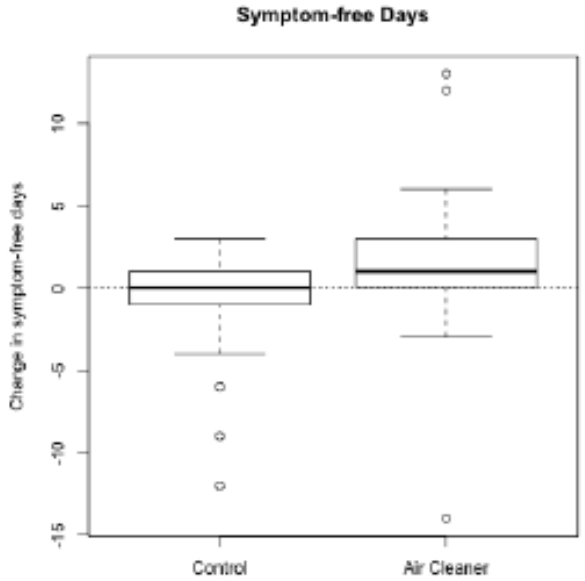
- more than 2 variables because the real world is multivariate
- show as much data on a plot as you can
- example

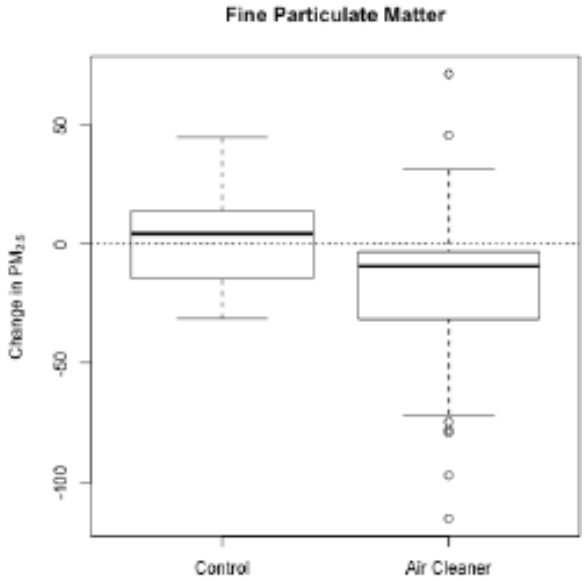
https://www.sfu.ca/cmns/courses/2011/325/Readings/Tufte-Analytical%20Design.pdf

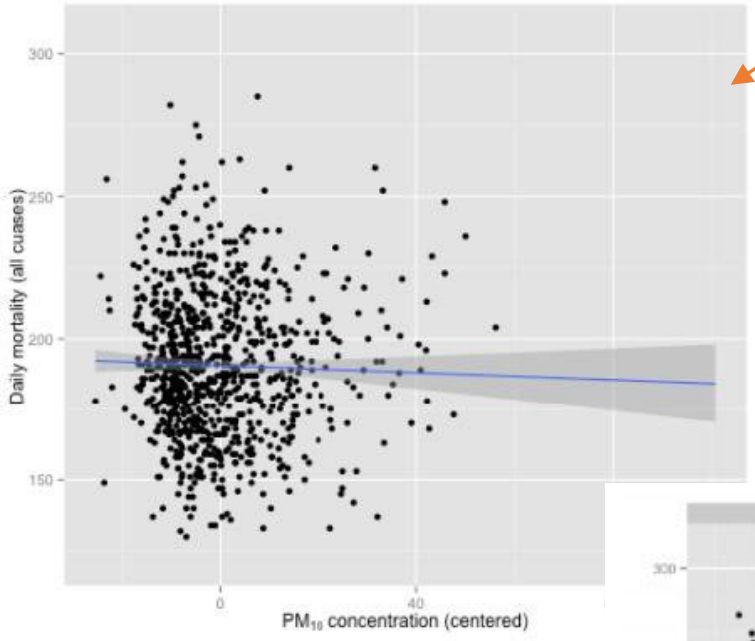
Show Comparisons



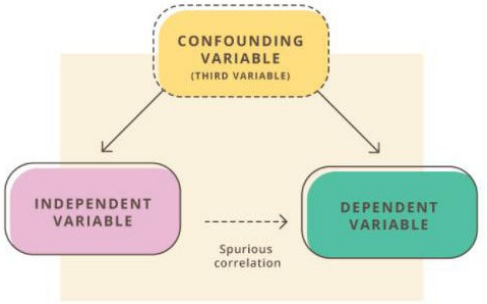
Show causality, mechanism

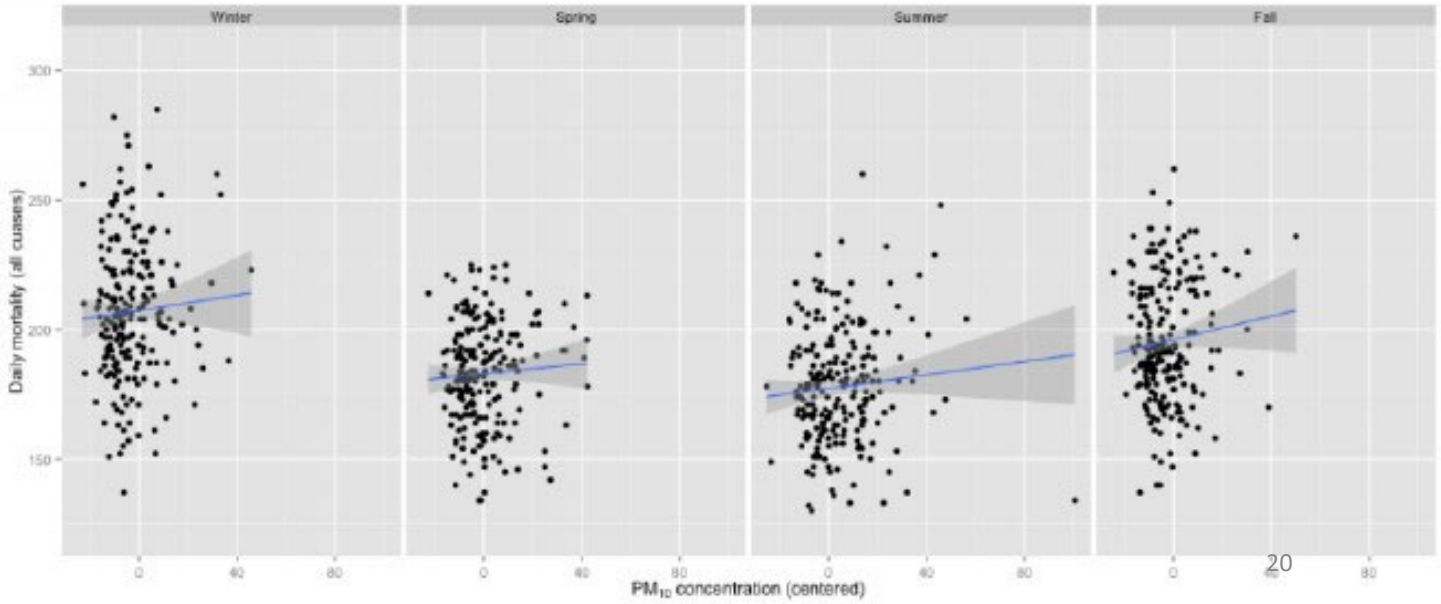






Slightly negative relationship between pollution and mortality
– when split up by season, the relationships are all positive → season = confounding variable





Show Multivariate Data

Principles of Analytical Design

Principle 4: Integration of evidence

– use as many modes of evidence/displaying evidence as possible (modes of data presentation)
– integrate words/numbers/images/diagrams (information rich)
– analysis should drive the tool.

Principle 5: Describe/document evidence with appropriate labels/ scales/ sources

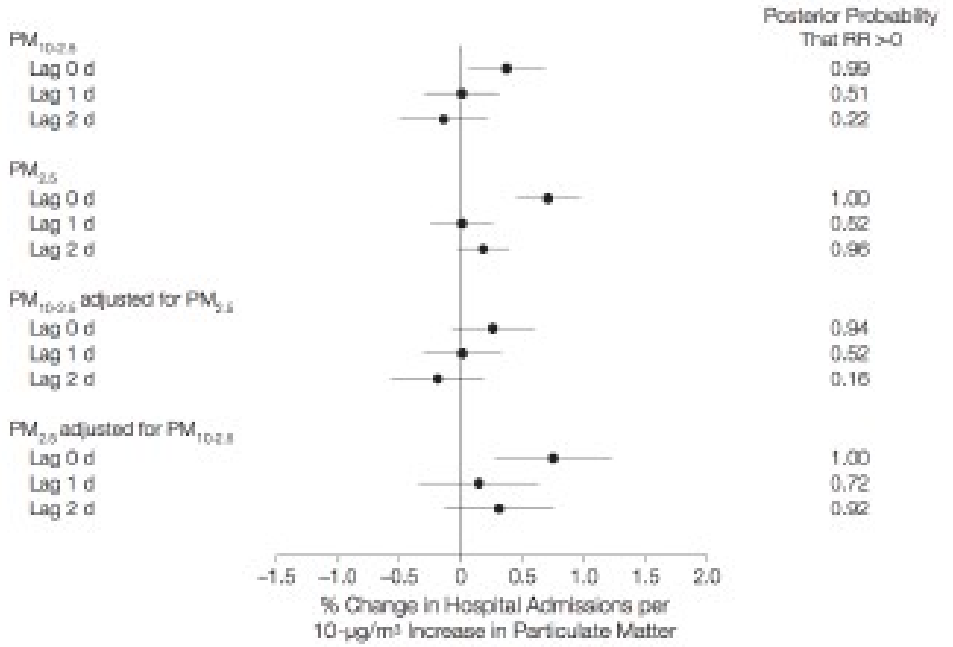
–Provide a detailed title, indicate the authors and sponsors, document the data sources, show complete measurement scales, point out relevant issues.
- add credibility to that data graphic

Principle 6: Content is the most important

– analytical presentations ultimately stand/fall depending on quality / relevance/ integrity of content.

Integrate Different Modes of Evidence

Figure 2. Percentage Change in Emergency Hospital Admissions Rate for Cardiovascular Diseases per a 10-μg/m³ Increase in Particulate Matter



Estimates are on average across 108 counties. $PM_{2.5}$ indicates particulate matter is 2.5 μm or less in aerodynamic diameter; PM_{10}, particulate matter is 10 μm or less in aerodynamic diameter; $PM_{10-2.5}$, particulate matter is greater than 2.5 μm and 10 μm or less in aerodynamic diameter; RR, relative risk. Error bars indicate 95% posterior intervals.

Ways of Doing EDA

	UNIVARIATE	MUTIVARIATE

Graphical	• **Quantitative Variable**:
	  • Histogram
	  • Boxplots
	  • Normal QQ-plot
	• **Categorical Variable**: Bar Charts
	• Time data – Line Plot
		• **One Categorical and One Quantitative Variable:**
		  • Side-by-side Boxplots
		• **Two or More Categorical Variables:**
		  • Grouped Bar Chart
		• **Two or More Quantitative Variables:**
		  • Scatterplot
		  • Correlation Heatmap
		  • Pairplot
		• Missing Data Detection

Non-Graphical	• **Categorical Variable:** tabular representation of frequency (*or relative frequency*)
	• **Quantitative Variable:**
	  • Location (*mean, median)*
	  • Spread (*IQR, std dev, range*)
	  • Modality (*mode*)
	  • Shape (*skewness, kurtosis*)
	  • Outliers
	• Missing Data Detection
		• **One Categorical and One Quantitative Variable:** ***standard univariate nongraphical statistics for the quantitative variables separately for each level of the categorical variable.***
		  • Mean
		  • Median
		  • Range and Spread measures
		• **Two or More Quantitative Variables:**
		  • Correlation
		  • Covariance
		  • Descriptive stat per
		• Missing Data Detection

Prerequisite for EDA

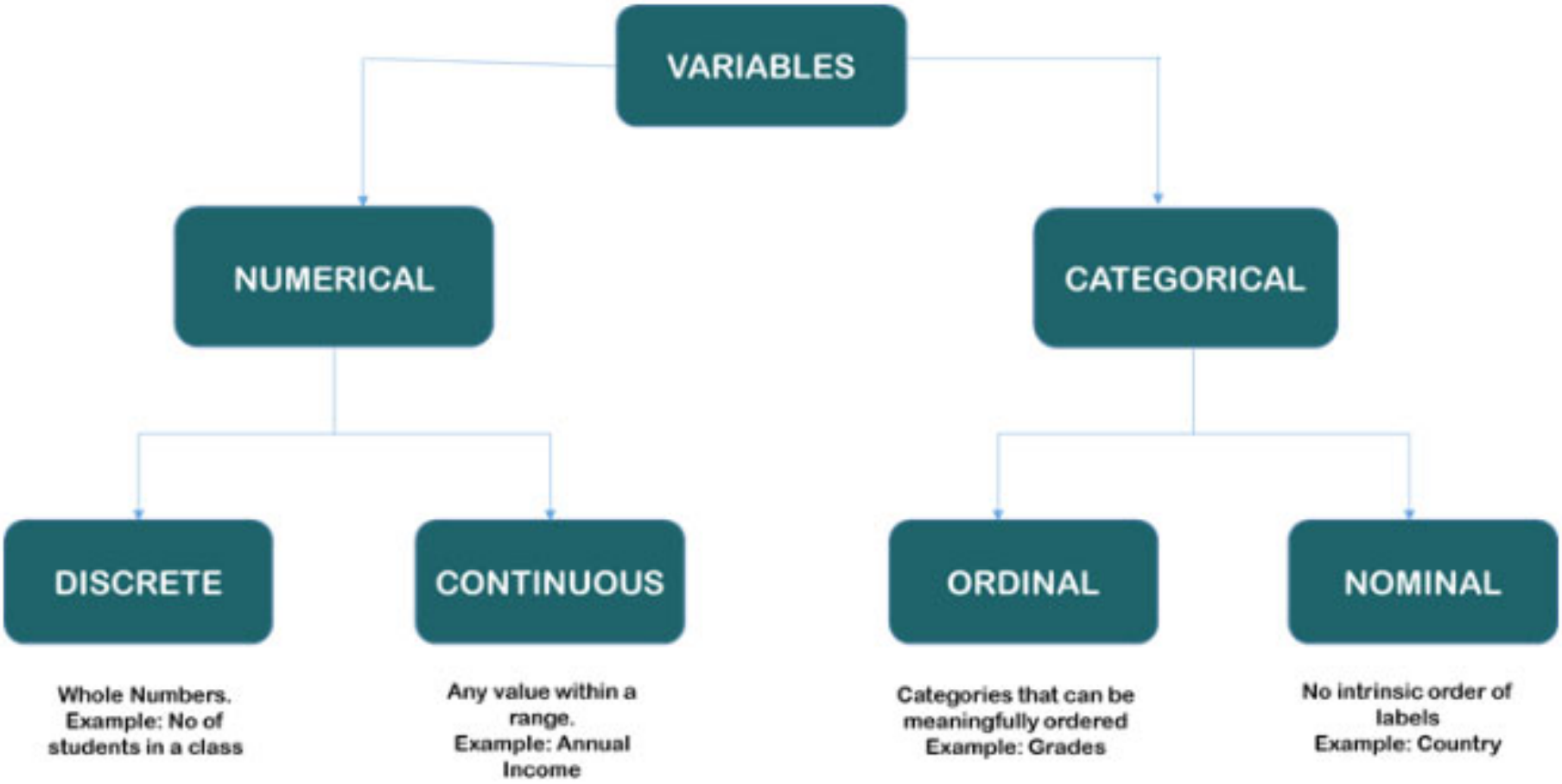
Type of Data

Data comes in various forms but can be classified into **TWO** main groups: **structured data** and **unstructured**.

- **Structured data** is data which is a form of data which has a high degree or organization such as **numerical** or **categorical** data.
  - e.g. Temperature, phone numbers, gender
- **Unstructured data** is data in a form which doesn't explicitly have structure.
  - e.g. photos, images, audio, language text and many others.

https://datascienceguide.github.io/exploratory-data-analysis

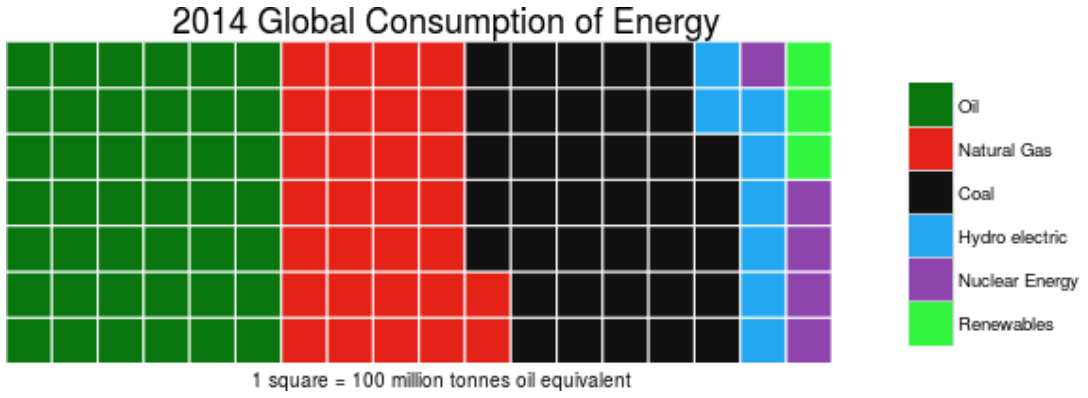
Classification of Variables



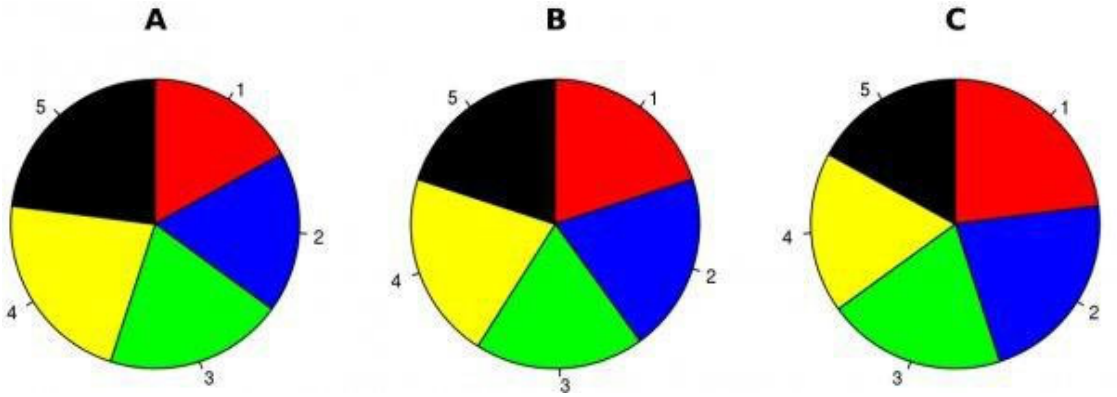
Categorical Variables

Categorical variables can also be **nominal** or **ordinal**.

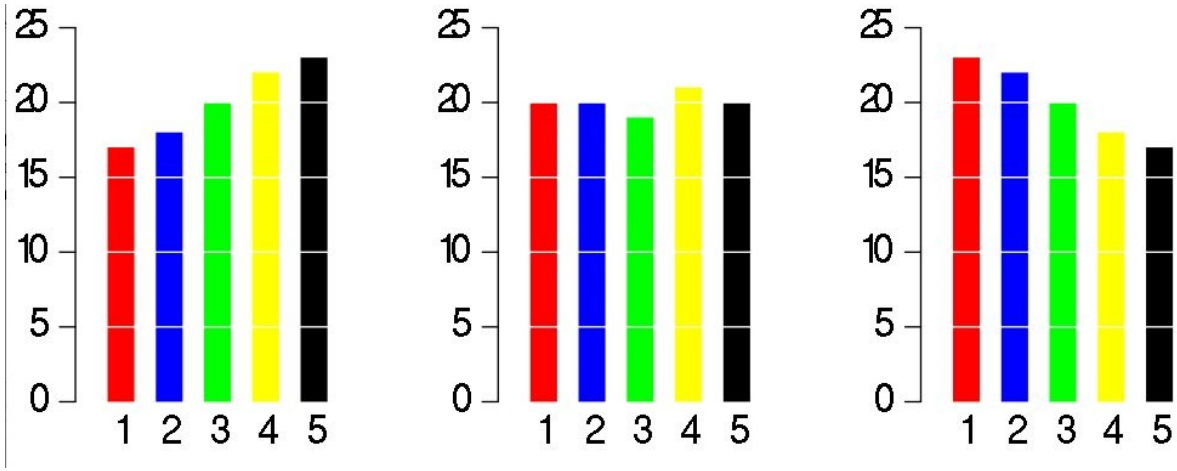
- **Nominal data** has no intrinsic ordering to the categories.
  - e.g. gender (Male, Female, Other) has no specific ordering.
- **Ordinal data** as clear ordering such as three settings on a toaster (high medium and low).
- A frequency table (count of each category) is the common statistic for describing categorical data of each variable, and a bar chart or a waffle chart (shown below) are two visualizations which can be used.



While a **pie chart** is a very common method for representing **categorical variables**, it is not recommended since it is very **difficult for humans to understand angles**.

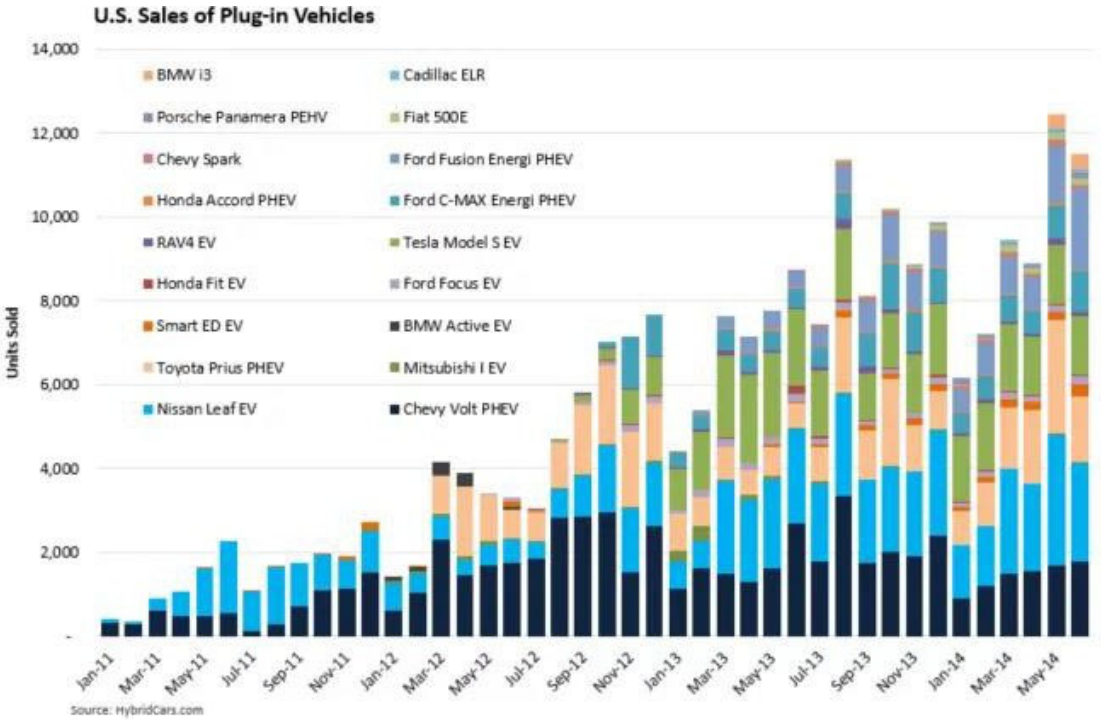


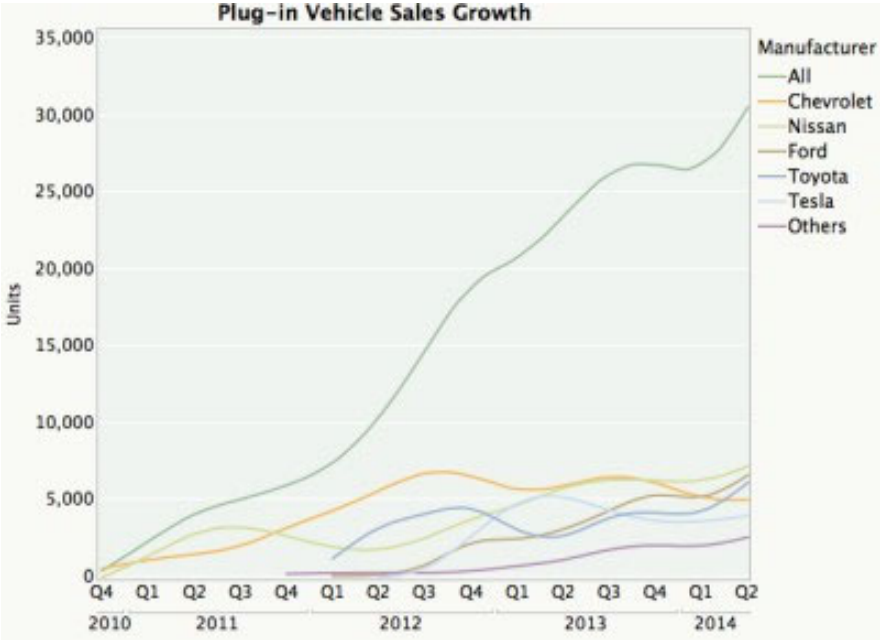
The charts **look identical**, and it takes more than a couple of seconds to understand the data. Now compare this to the corresponding **bar chart**:



A reader an instantly compare the 5 variables.

Less is More





This visualization is very complicated and difficult to understand.

When dealing with multi-categorical data, avoid using stacked bar charts.

Numeric Variables

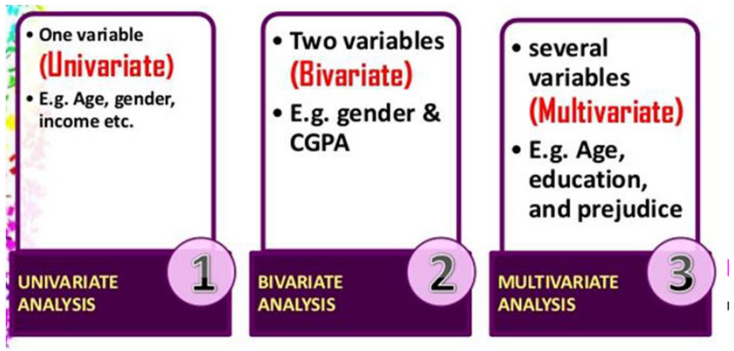
Numeric or **continuous** variables can be any value within a finite or infinite interval (e.g. include temperature, height, weight)

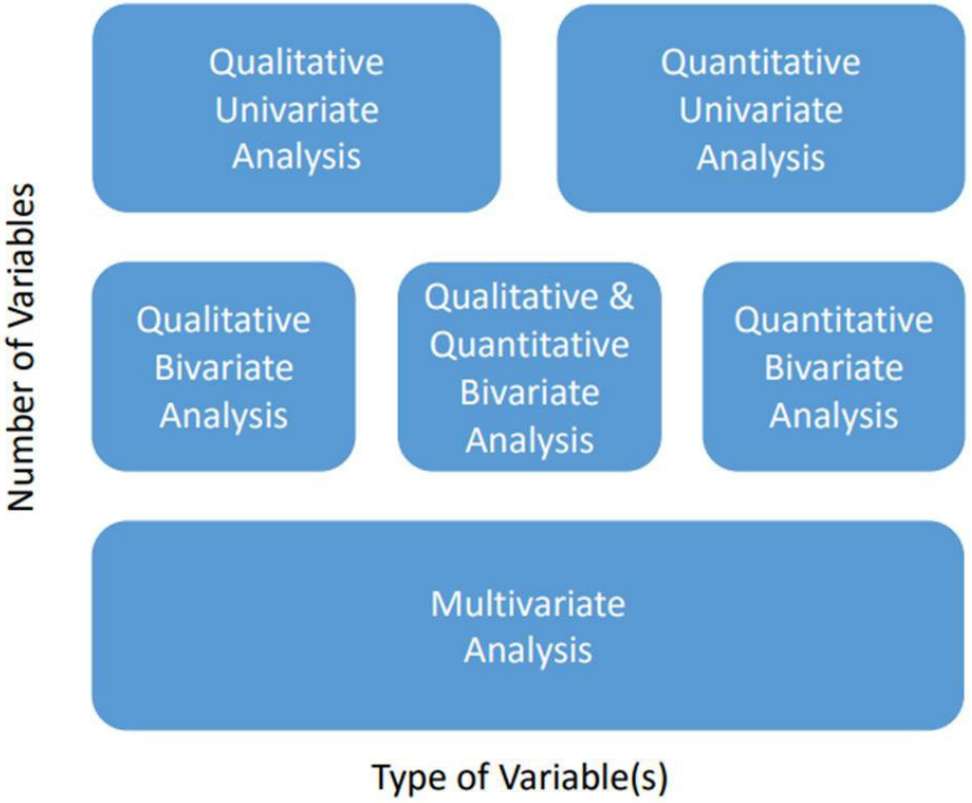
There are 2 types of numeric variables are **interval** and **ratios**.

- **Interval variables** have numeric scales and the same interpretation throughout the scale, but do not have an absolute zero.
  - e.g. temperature, grading in exam, IQ scores, calendar dates
- **Ratio variables** is interval data with a natural zero and can be added, subtracted, multiplied or divided.
  - e.g. height, weight, marks in exam, distance, salary

Types of Data Visualizations

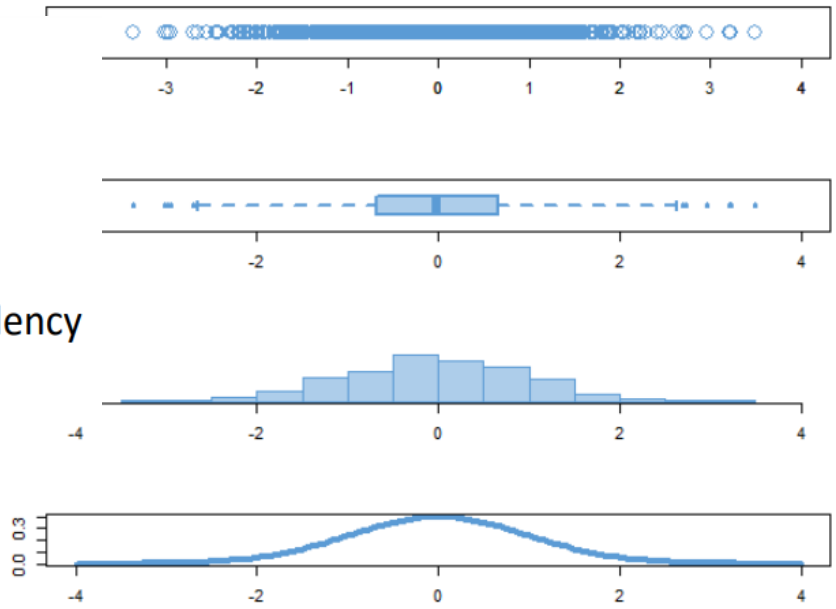
- Type of variables
  - Qualitative
  - Quantitative
- Number of variables
  - Univariate
  - Bivariate
  - Multivariate





Univariate Analysis

- One variable
- Qualitative
  - Frequency
- Quantitative
  - Central tendency
  - Dispersion



The two visualizations used to describe univariate (1 variable) data is the **box plot** and the **histogram**.

The box plot can be used to show the minimum, maximum, mean, median, quantiles and range.

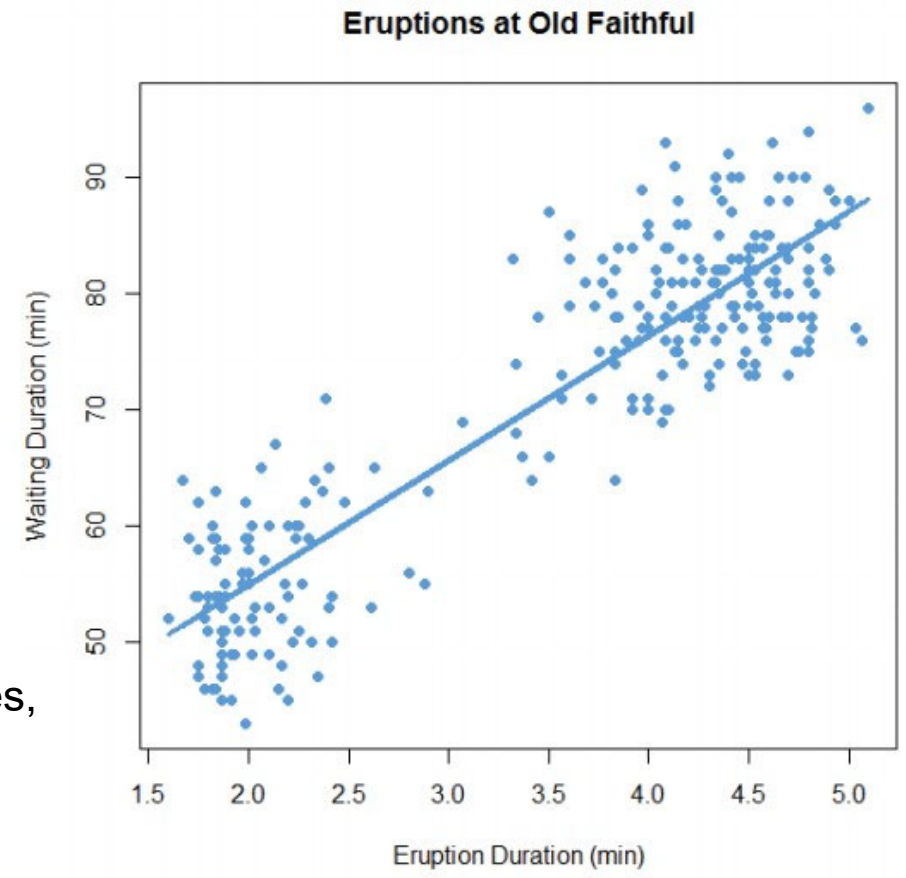
The histogram can be used to show the count, mode, variance, standard deviation, coefficient of deviation, skewness and kurtosis.

Bivariate Analysis

- Qualitative
  - Joint frequency
- Quantitative
  - Two variables
    - Predictor
    - Outcome
  - Measures
    - Covariance
    - Correlation

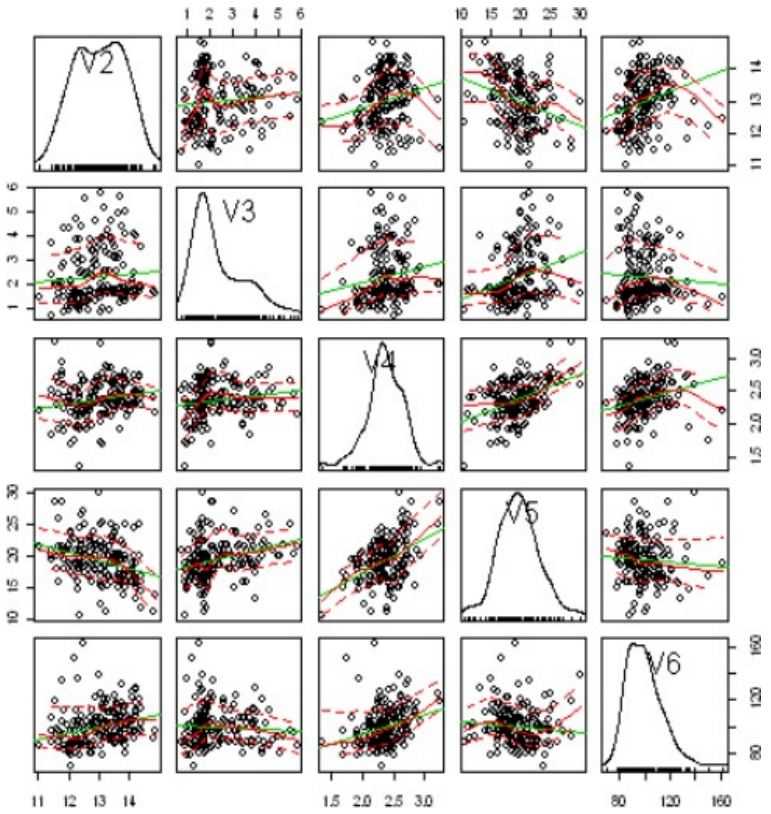
When plotting the relation between two variables, one can use a **scatter plot**.

If the data is time series or has an order, a **line chart** can be used.



Multivariate Analysis

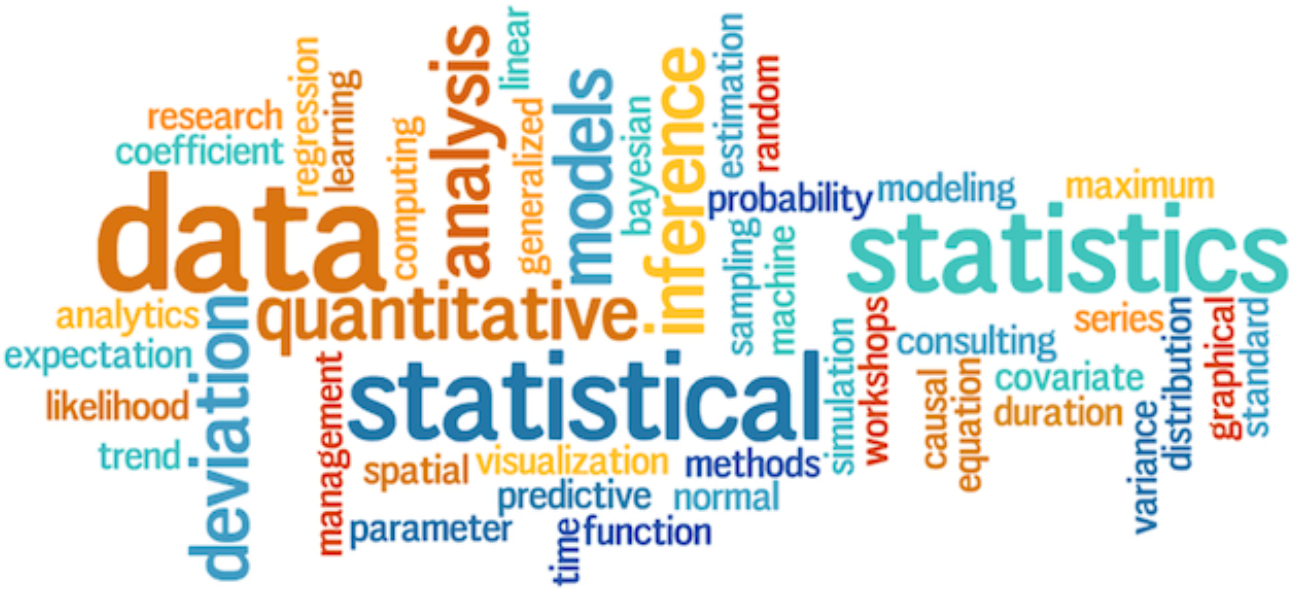
- Multivariate analysis uses **two or more variables** and analyzes which, if any, are correlated with a specific outcome.
- **The goal** is to determine which variables influence or cause the outcome.
- **Multiple regression analysis** is the most common method used in multivariate analysis to find correlations between data sets, but many others, such as logistic regression and multivariate analysis of variance, are also used.
- For visualization, use a **scatter plot** that shows the relation between each variables.



A Little Book of R For Multivariate Analysis

https://media.readthedocs.org/pdf/little-book-of-r-for-multivariate-analysis/latest/little-book-of-r-for-multivariate-analysis.pdf

Textual Data



A **word cloud** (or tag cloud) depicts **text data**, typically by **sizing each word** proportionally to its **frequency** within the text.

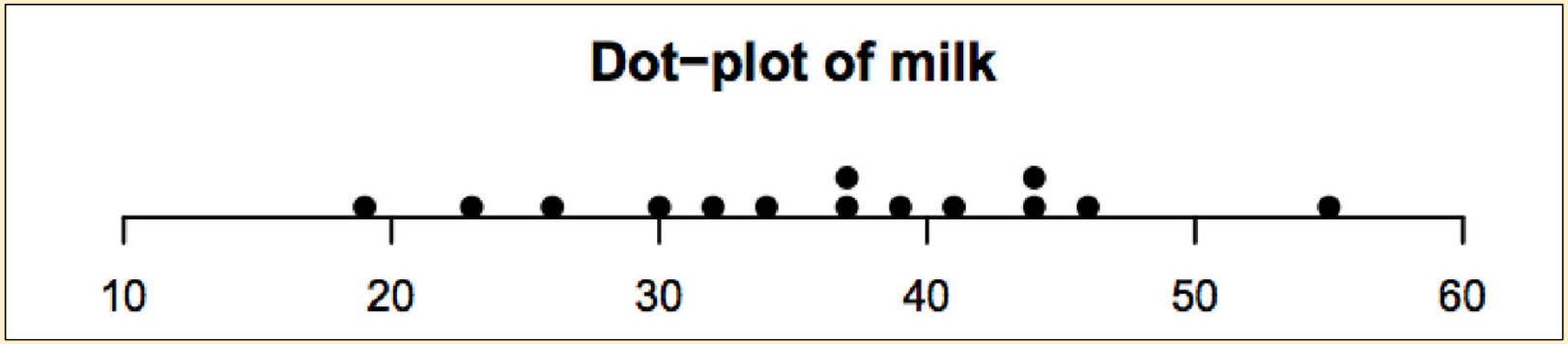
To remove the words which add noise to the dataset, the documents can be grouped using **Topic modeling** and only the important words can be displayed.

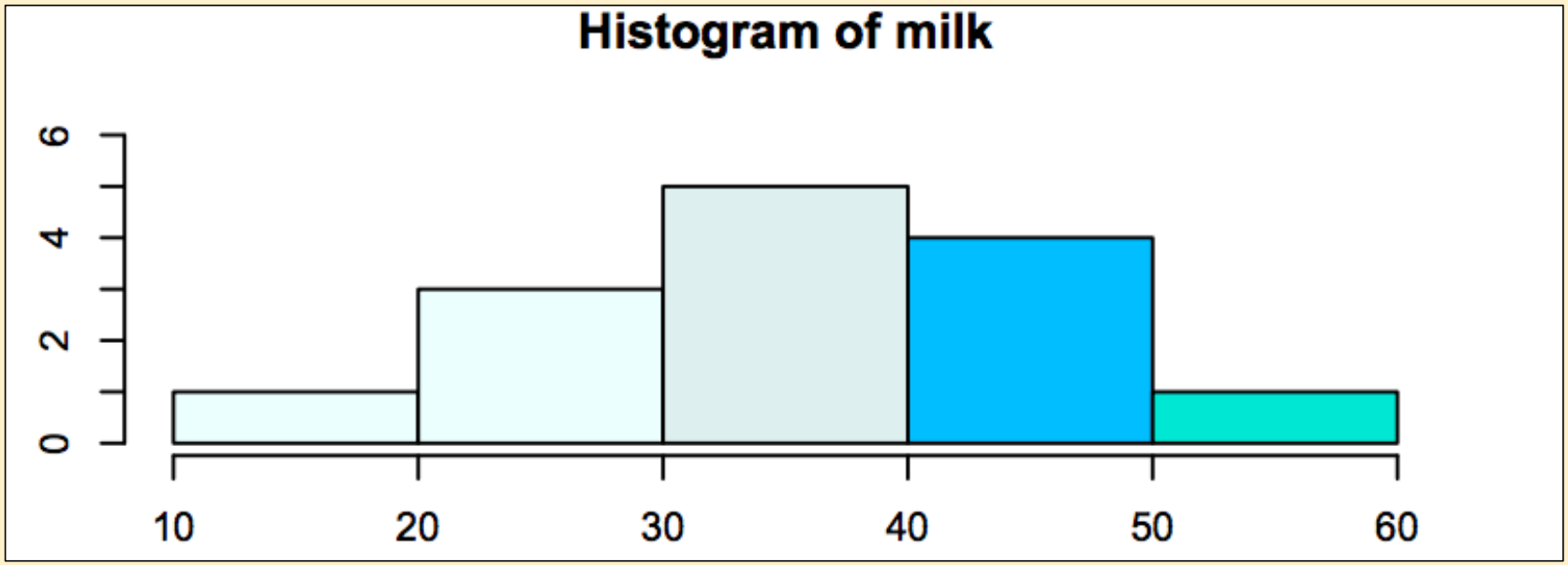
Milk Production

- Milk data: milk yields (lbs/day) were collected form a herd of 14 cows in a single day.
- Data: 44,55,37,32,37,26,23,41,34,19,30,39,46,44
- How to represent this data in R?
- What is the best way to summarize this dataset?

```
> milk <- c(44,55,37,32,37,26,23,41,34,19,30,39,46,44)
```

Visual Summaries





Measure of Centers

- Mean
- Median
- Mode

```
> mean (milk)
[1] 36.21429

> sort(milk)
Sort the data: 19 23 26 30 32 34 37 37 39 41 44 44 46 55

> median(milk)
[1] 37
```

Quartiles

Milk yield:
19 23 26 30 32 34 37 | 37 39 41 44 44 46 55

Five Number Summary and Boxplot

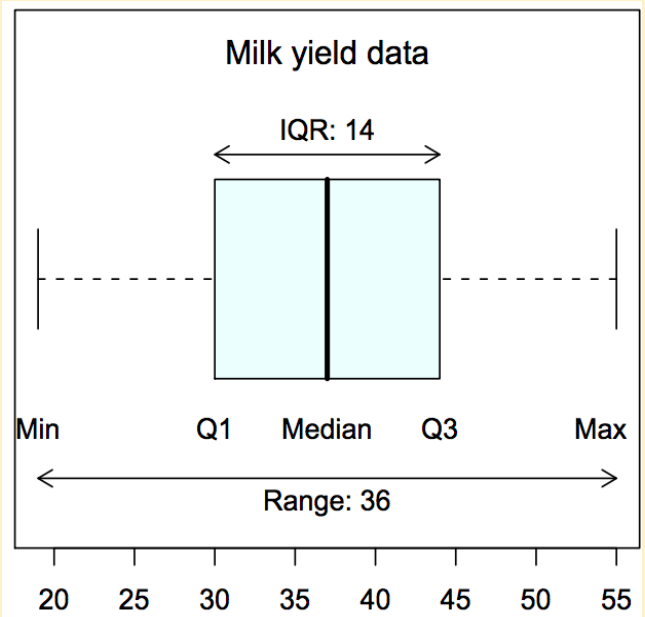
- **Five number summary** – minimum, maximum, median, and the quartiles.
- A boxplot is a visual representation of the five-number summary.

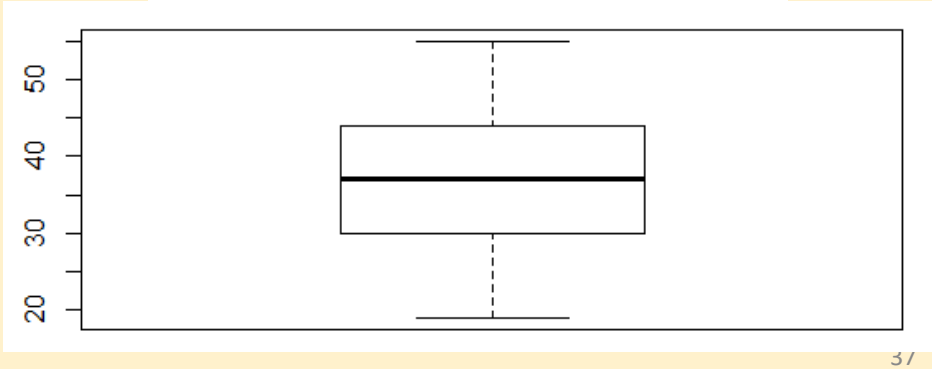
```
> fivenum(milk)
> summary(milk)
> boxplot(milk)
```

```
> fivenum(milk)
[1] 19 30 37 44 55
```

Returns Tukey's five number summary (minimum, lower-hinge (Q1), median, upper-hinge (Q3), maximum) for the input data.

```
> summary (milk)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  19.00   30.50   37.00   36.21   43.25   55.00
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  19.00   30.50   37.00   36.21   43.25   55.00
```





Stem-Leaf Display

> stem (milk)

```
The decimal point is 1 digit(s) to the right of the |

 1 | 9
 2 | 36
 3 | 024779
 4 | 1446
 5 | 5
```

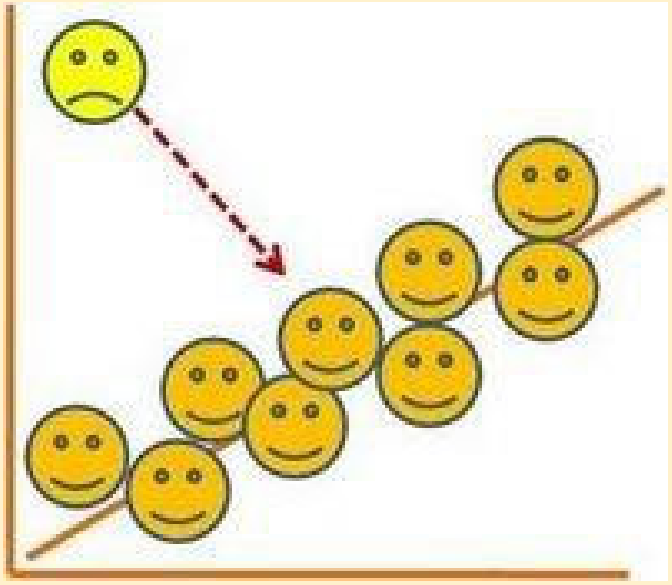
Histograms are useful with larger datasets, stem-leaf displays with smaller data sets.

Outliers

An **outlier** is a data point which differs so much from the rest of the data that it doesn't seem to belong.

Possible reasons:

- typographical error
- problem with the experimental protocol
- special circumstances (e.g. an abnormally high value on a medical test might indicate the presence of a disease)



Detecting Outlier in Milk Example

- $Q_1 = 30$, $Q_3 = 44$, $IQR = 44 - 30 = 14$
- Fences, $1.5 * 14 = 21$

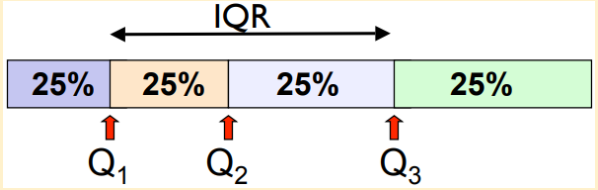
$$\text{lower fence} = 30 - 21 = 9$$

$$\text{upper fence} = 44 + 21 = 65$$

- Outliers?
  1. smallest data point (min) $= 19 > 9$
  2. largest (max) $= 55 < 65$
  3. no oultier

Measures of Dispersion

- Range
- Interquartile range
- Variance
- Standard deviation
- Coefficient of variation



Range: maximum - minimum
Milk yield: range is 55 − 19 = 36
IQR: Inter Quartile Range = Q3 − Q1
Spread in the central "body" of the distribution
Milk yield: IQR = 44 − 30 = 14

Dispersion as "Deviation from the Mean"

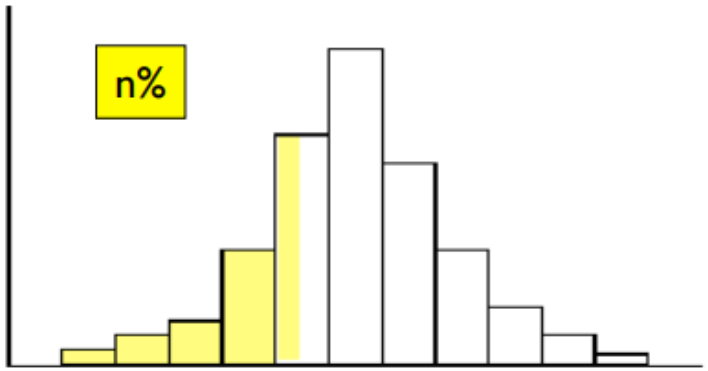
The **variance**, **standard deviation**, and **coefficient of variation** are all related.
Based on deviations from the mean, is the signed distance of an observation from the mean.

deviation = value of observation − mean

Observations **greater than the mean have positive deviations** while those **less than the mean have negative deviations**.
Formula (for the ith observation): $y_i - \bar{y}$
Ex: first cow has deviation 44 − 36.2 = +7.8, cow with data 19 has deviation 19 − 36.2 = −17.2.

Percentiles (aka Quantiles)

In general the n^{th} percentile is a value such that n% of the observations fall at or below or it.



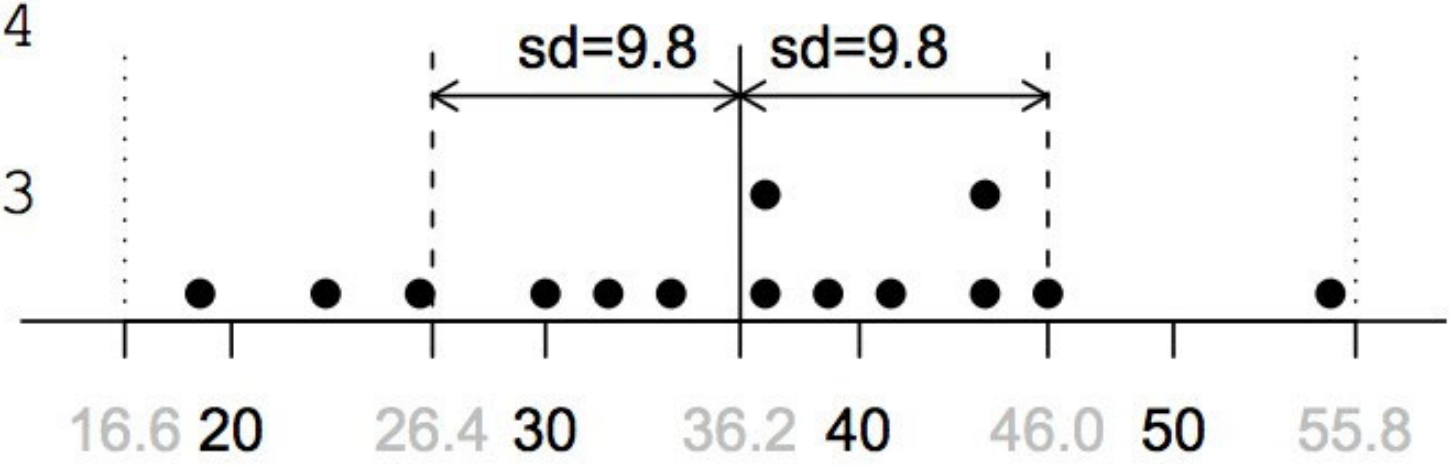
Q_1 = 25th percentile

Median = 50th percentile

Q_2 = 75th percentile

Standard Deviation

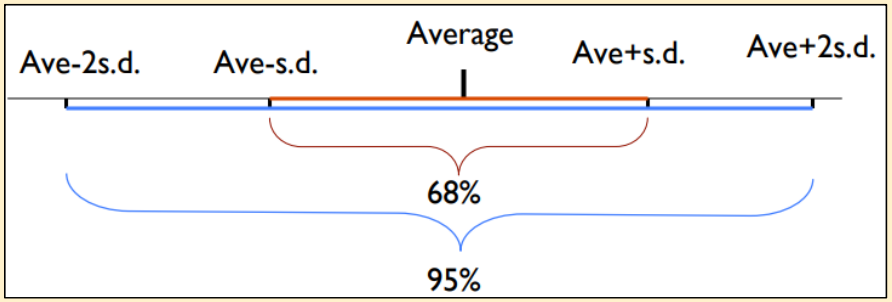
```
> mean(milk)
[1] 95.25824
> sd(milk)
[1] 9.760033
```



Empirical Rule

For many variables (especially those that are nearly symmetric and bell-shaped), the following empirical rule is often a very good approximation.

- About 68% of the observations are within 1 SD of the mean.
- About 95% of the observations are within 2 SDs of the mean.



Coefficient of variation

It is the relative variation CV = std dev / mean ($s / \bar{y}$)

It is dimensionless.

Milk data: CV = 9.8/36.2 = 0.27. It means the typical deviation from the mean is about 27% of the mean.

Demo

Console ~/RDemo/

```
> getwd()
[1] "C:/Users/Salimah/Documents/RDemo"
> # Load CSV data
> movies <- read.csv("Movies.csv")
>
> genres <- read.csv("Genres.csv")
>
> # Peek at data
> head(movies)
                 Title Year Rating Runtime Critic.Score Box.Office
1  The Whole Nine Yards 2000      R      98           45       57.3
2             Gladiator 2000      R     155           76      187.3
3      Cirque du Soleil 2000      G      39           45       13.4
4              Dinosaur 2000     PG      82           65      135.6
5     Big Momma's House 2000  PG-13      99           30        0.5
6 Gone in Sixty Seconds 2000  PG-13     118           24      101.0
>
> head(genres)
                 Title  Genre Year Rating Runtime Critic.Score Box.Office
1 The Whole Nine Yards  Crime 2000      R      98           45       57.3
2 The Whole Nine Yards Comedy 2000      R      98           45       57.3
3     Cirque du Soleil  Drama 2000      G      39           45       13.4
4     Cirque du Soleil Family 2000      G      39           45       13.4
5             Gladiator Action 2000      R     155           76      187.3
6             Gladiator  Drama 2000      R     155           76      187.3
>
```

2 files to load

Univariate Analysis

```
Console ~/RDemo/
>
> # Univariate statistics for a quantitative variable
> # (i.e. one numeric variable)
>
> # Analyze central tendancy (location)
> mean(movies$Runtime)
[1] 104.4052
>
> median(movies$Runtime)
[1] 101
>
> # Analyze the dispersion (spread)
> min(movies$Runtime)
[1] 38
>
> max(movies$Runtime)
[1] 219
>
> range(movies$Runtime)
[1]  38 219
>
> diff(range(movies$Runtime))
[1] 181
>
> quantile(movies$Runtime)
  0%  25%  50%  75% 100%
  38   93  101  113  219
>
> quantile(movies$Runtime, 0.95)
95%
135
>
> IQR(movies$Runtime)
[1] 20
>
> var(movies$Runtime)
[1] 284.4487
>
```

```
Console ~/RDemo/
>
> sd(movies$Runtime)
[1] 16.86561
>
> # Summarize a quantitative variable
> summary(movies$Runtime)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   38.0    93.0   101.0   104.4   113.0   219.0
>
```

Bivariate Analysis

Console ~/RDemo/

```
>
> # Bivariate statistics for two qualitative variables
> # (i.e. two categorical variables)
> table(genres$Genre, genres$Rating)

                G  PG PG-13   R
  Action        2  70   311 229
  Adventure    44 179   209  64
  Animation    43 111     8   6
  Biography     0  27    73  93
  Comedy       45 258   472 506
  Crime         0   9   141 328
  Documentary  27  73    78  65
  Drama        12 136   586 836
  Family       38 181    10   1
  Fantasy       6  51   115  43
  History       3  12    36  35
  Horror        0   3    71 195
  Music         5  31    81  59
  Musical       0  11    20   6
  Mystery       0   6   102 136
  Sci-Fi        0   7   119  72
  Sport         4  36    62  19
  Thriller      0   2   167 324
  War           1   0    19  31
  Western       0   4     6  10
>
> # Bivariate statistics for two quantitative variables
> # (i.e. two numeric variables)
>
> # Covarience
> cov(movies$Runtime, movies$Box.Office)
[1] 381.624
>
> cov(movies$Critic.Score, movies$Box.Office)
[1] 289.6335
>
```

```
Console ~/RDemo/
>
> # Correlation coefficients
> cor(movies$Runtime, movies$Box.Office)
[1] 0.347748
>
> cor(movies$Critic.Score, movies$Box.Office)
[1] 0.1608324
>
> # Bivariate statistics for both a qualitative and quantitative variable
> # (i.e. one categorical and one numeric variable)
> tapply(movies$Box.Office, movies$Rating, mean)
       G       PG    PG-13        R
55.47561 56.40439 54.56134 22.26118
>
> tapply(genres$Box.Office, genres$Genre, mean)
     Action   Adventure   Animation   Biography      Comedy       Crime Documentary       Drama
  76.530806  101.745110   96.603311   26.500308   40.860973   34.320142    6.268575   24.740296
     Family     Fantasy     History      Horror       Music     Musical     Mystery      Sci-Fi
  68.339200   93.251211   24.181583   27.932895   21.978918   37.172776   40.328661   86.874763
      Sport    Thriller         War     Western
  27.739240   38.523364   26.474298   36.146105
>
> # Summarize entire table
> summary(movies)
                  Title          Year        Rating        Runtime       Critic.Score
 Camp                 :   2   Min.   :2000   G    :  93   Min.   : 38.0   Min.   :  0.00
 Frozen               :   2   1st Qu.:2004   PG   : 497   1st Qu.: 93.0   1st Qu.: 26.00
 The Other Woman      :   2   Median :2008   PG-13:1225   Median :101.0   Median : 49.00
 (500) Days of Summer:   1   Mean   :2008   R    :1423   Mean   :104.4   Mean   : 49.68
 (Untitled)           :   1   3rd Qu.:2011                3rd Qu.:113.0   3rd Qu.: 74.00
 10 Items or Less     :   1   Max.   :2015                Max.   :219.0   Max.   :100.00
 (Other)              :3229
   Box.Office
 Min.   :  0.0002
 1st Qu.:  1.0000
 Median : 16.1000
 Mean   : 40.6756
 3rd Qu.: 51.4750
 Max.   :760.5000
```

Exploratory Data Analysis with R package

	CRAN			GitHub				
package	downl.	debut	age	stars	commits	contrib.	issues	forks
arsenal	39234	2016-12-30	2y 6m	59	637	3	200	4
autoEDA	-	-	-	41	20	1	4	12
DataExplorer	82624	2016-03-01	3y 4m	235	187	2	121	44
dataMaid	23972	2017-01-02	2y 6m	68	473	2	45	18
dlookr	13268	2018-04-27	1y 2m	35	54	3	9	12
ExPanDaR	5713	2018-05-11	1y 2m	32	197	2	3	14
explore	808	2019-05-16	0y 1m	15	114	1	1	0
exploreR	8112	2016-02-10	3y 5m	1	1	1	0	0
funModeling	54232	2016-02-07	3y 5m	58	126	2	13	18
inspectdf	3252	2019-04-24	0y 2m	117	200	2	12	11
RtutoR	10502	2016-03-12	3y 3m	13	7	1	4	8
SmartEDA	5150	2018-04-06	1y 3m	4	4	1	1	2
summarytools	84737	2014-08-11	4y 11m	255	981	6	76	33
visdat	68978	2017-07-11	2y 0m	313	426	12	122	39
xray	8300	2017-11-22	1y 7m	63	33	4	10	5

https://arxiv.org/pdf/1904.02101.pdf

Exploratory Graphs

Purpose:

- To understand data properties
- To find patterns in data
- To suggest modeling strategies
- To "debug" analyses
- To communicate results.

Characteristics:

- They are made quickly
- A large number are produced
- The goal is to gain personal understanding
- Axes/legends are generally cleaned up later (appearances and presentation are aren't as important.)
- Color / size are primarily used for information.

One dimension

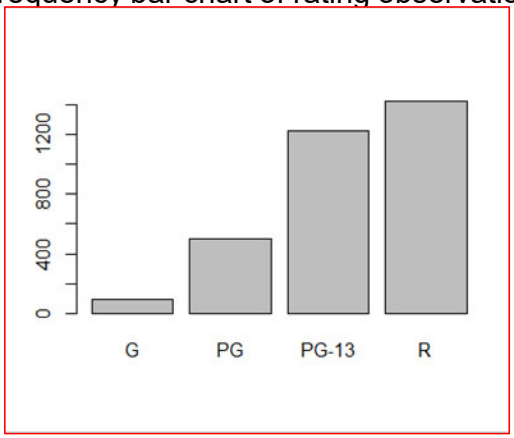
- ✓ Five-number summary
- ✓ Boxplots
- ✓ Histograms
- ✓ Density plot
- ✓ Barplot

Two dimensions

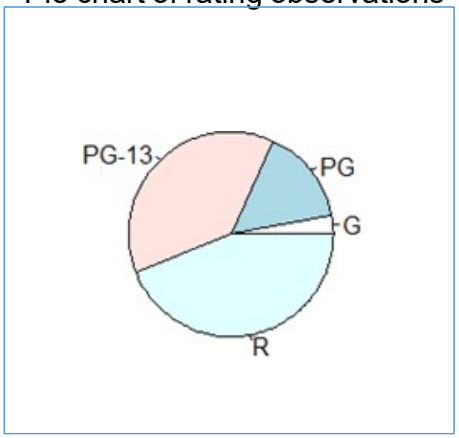
Multiple/overlayed 1-D plots (Lattice/ggplot2)
Scatterplots
Smooth scatterplots

Data Visualization

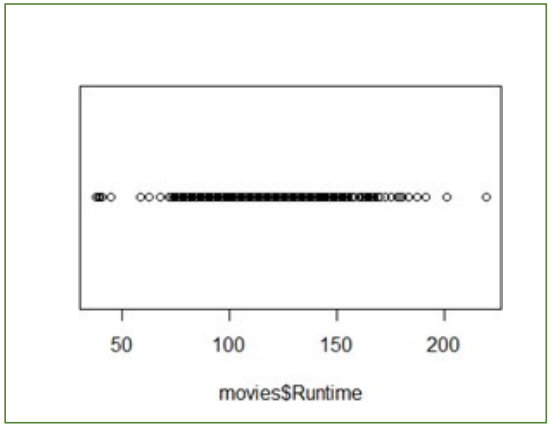
Frequency bar chart of rating observations



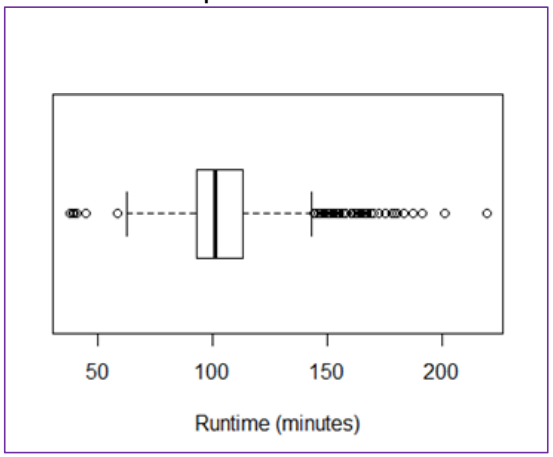
Pie chart of rating observations



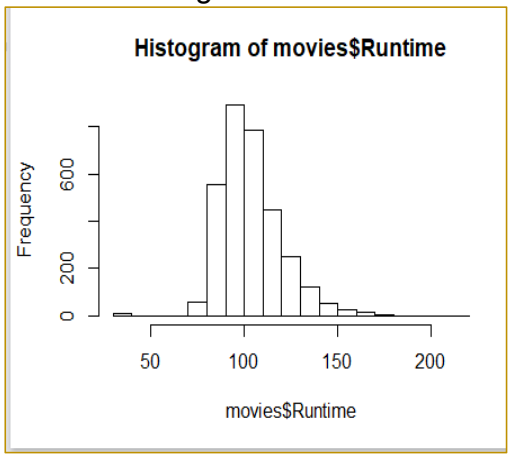
Dot plot of runtime



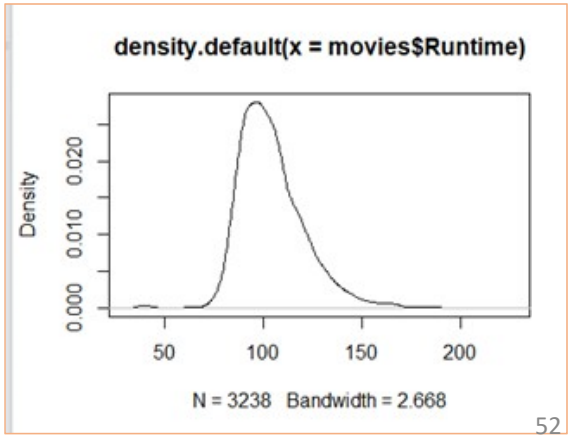
Boxplot of runtime



Histogram of runtime



Density plot of runtime



Plotting Systems in R

There are **THREE** common plotting systems available in R.

1. **{base}** which is in the default R installation.
2. **{lattice}** which is loaded using the *library(lattice)* command.
3. **{ggplot2}** which is loaded using the *library(ggplot2)* command.

Plotting occurs in **TWO** stages:
The base plotting system is very flexible and offers a high degree of control over plotting

- Creation of a plot
- Annotation of a plot (adding lines, points, text, legends)

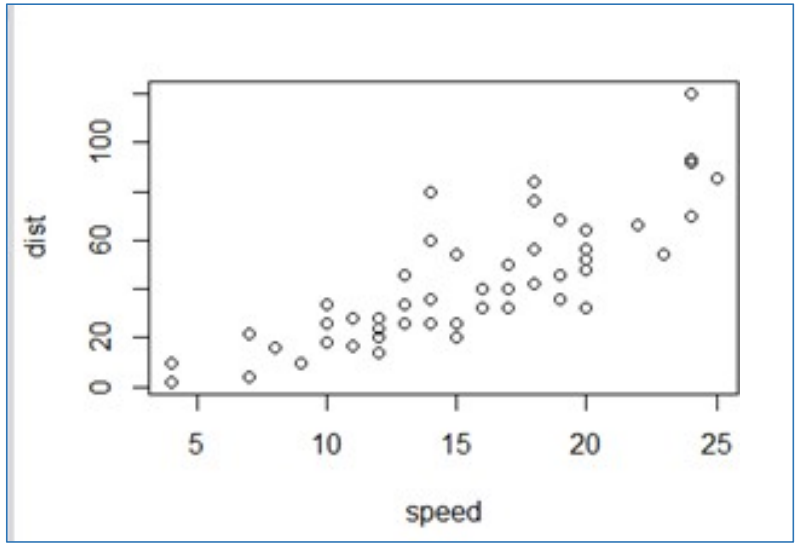
The Base Plotting System

- Convenient, mirrors how we think of building plots and analyzing data.
- Can’t go back once plot has started (i.e. to adjust margins); need to plan in advance.
- Difficult to "translate" to others once a new plot has been created (no graphical "language").
- Plot is just a series of R commands.

Base: "artist's palette" model

Base graphics are usually constructed piecemeal, with each aspect of the plot handled separately through a series of function calls; this is often conceptually simpler and allows plotting to mirror the thought process.

```
>
>
> library(datasets)
> data(cars)
> with(cars, plot(speed, dist))
>
```

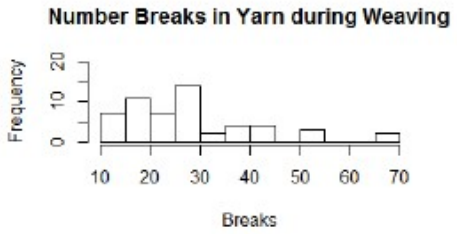


{base} graphics system

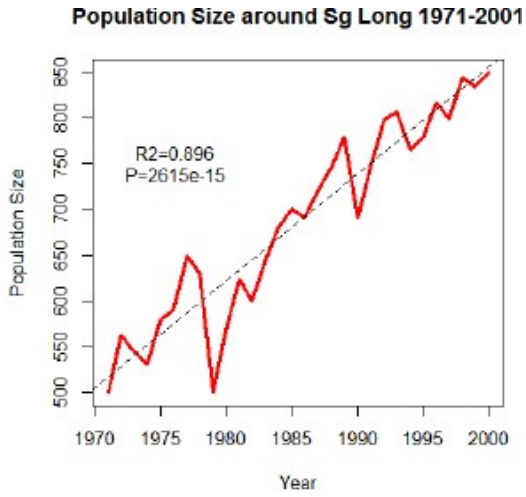
- Common main plotting functions.
  - `hist()`, `barplot()`, `boxplot()`, `plot()`

- They are usually annotated with separate functions. The following functions are typically used to add elements to a plot.
  - `points()`, `lines()`, `text()`

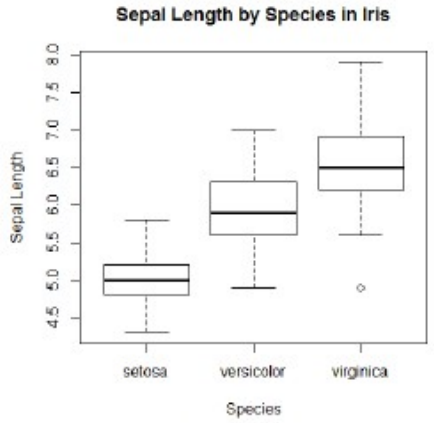
Plots



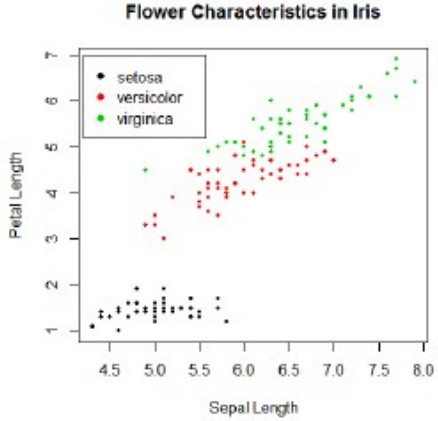
Basic Histogram



Line graph with regression

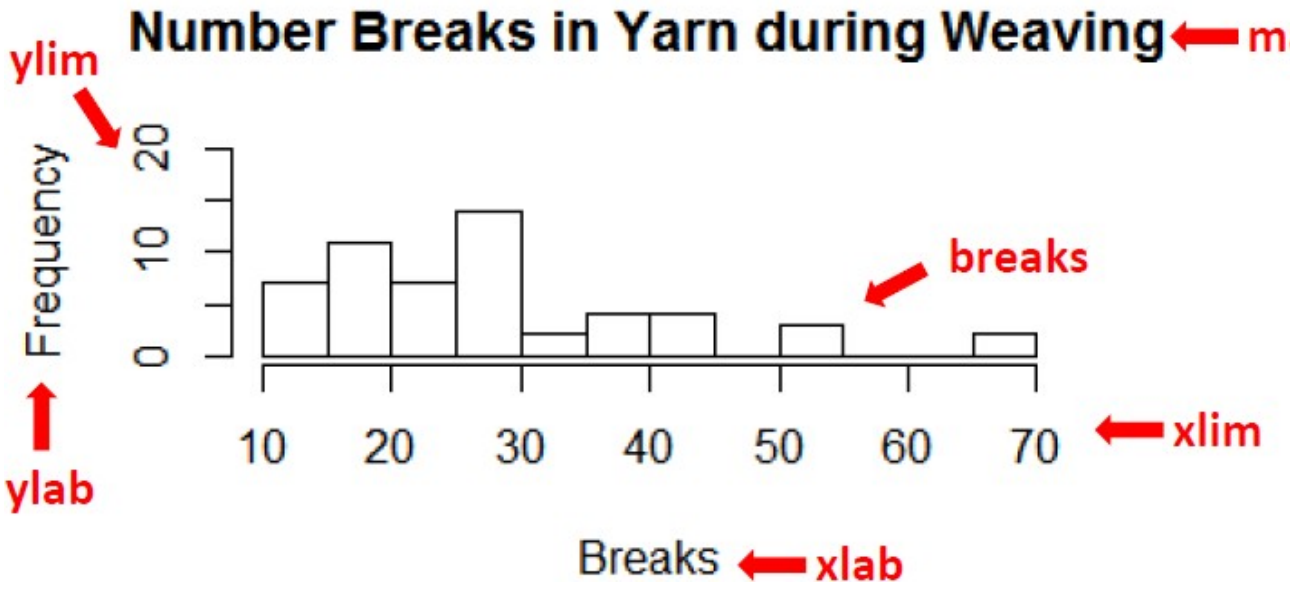


Basic Boxplot



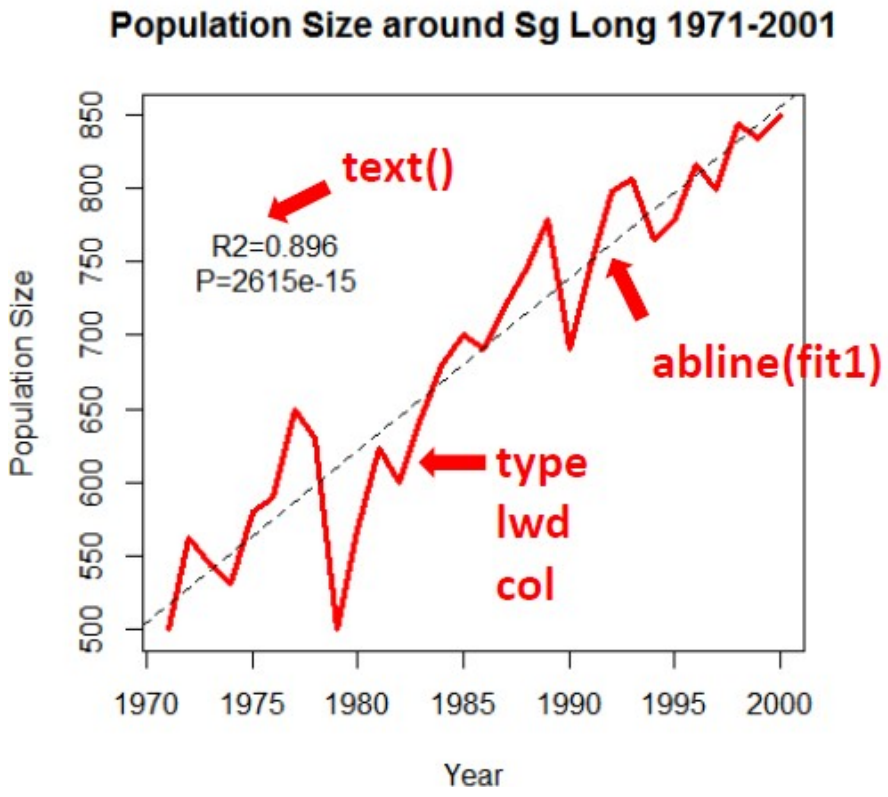
Scatterplot

Basic Histogram



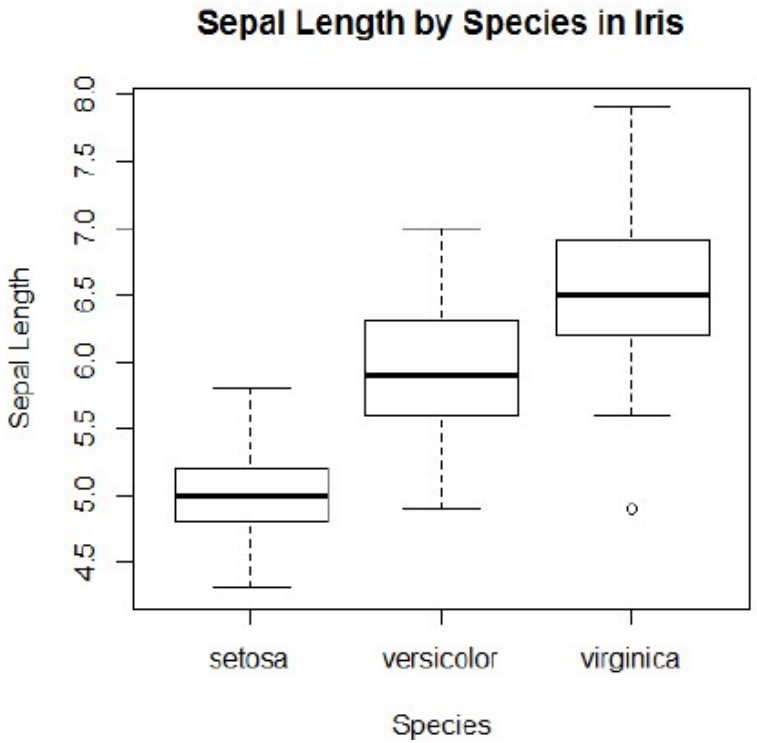

```
library(datasets)
hist(warpbreaks$breaks,
     breaks=20,
     xlab = "Breaks",
     main="Number
Breaks in Yarn during
Weaving",
     ylim = c(0,20))
```

Basic Line Plot



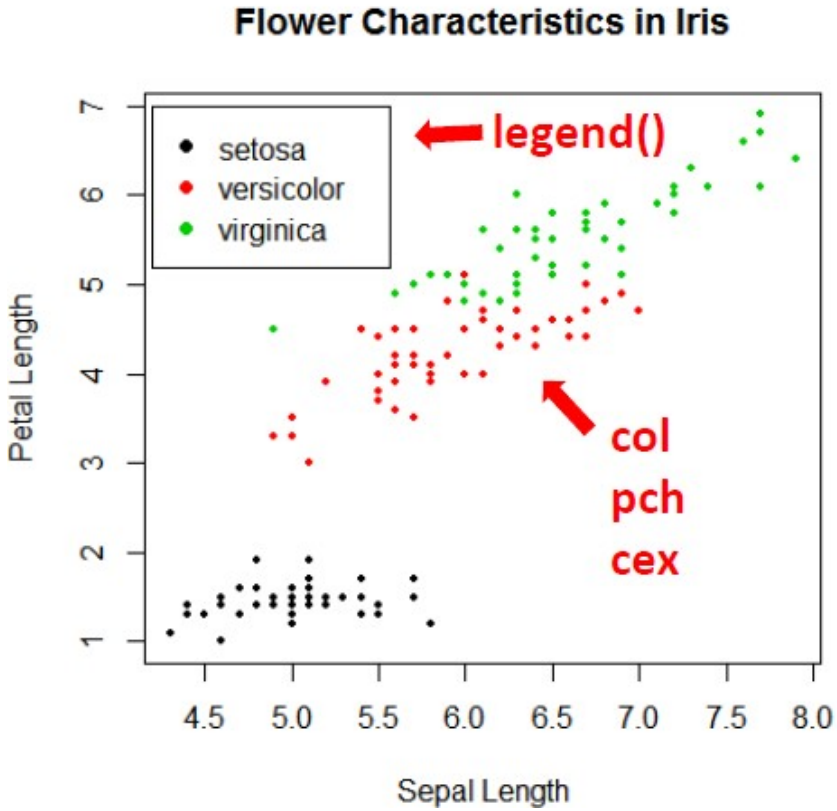

```
sglong<-read.table("sg_long_density.txt",
header=TRUE)
plot(sglong$Year, sglong$PopSize,
     type="l",
     col="red",
     lwd=3,
     xlab="Year",
     ylab="Population Size",
     main="Population Size around Sg Long
1971-2001")
fit1<-lm(PopSize~ Year, data=sglong)
abline(fit1, lty="dashed")
text(x=1976, y=750, labels=" R2=0.896\n
P=2615e-15")
```

Basic Boxplot




```
library(datasets)
boxplot(iris$Sepal.Length ~
iris$Species,
        ylab="Sepal Length",
        xlab="Species",
        main="Sepal Length by
Species in Iris")
```

Basic Scatterplot




```
Library(datasets)
plot(iris$Sepal.Length,
iris$Petal.Length,
     col=iris$Species,
     pch=16,
     cex=0.5,
     xlab="Sepal Length",
     ylab="Petal Length",
     main="Flower Characteristics
in Iris")
legend(x=4.2, y=7,
legend=levels(iris$Species),col=c(
1:3), pch=16)
```

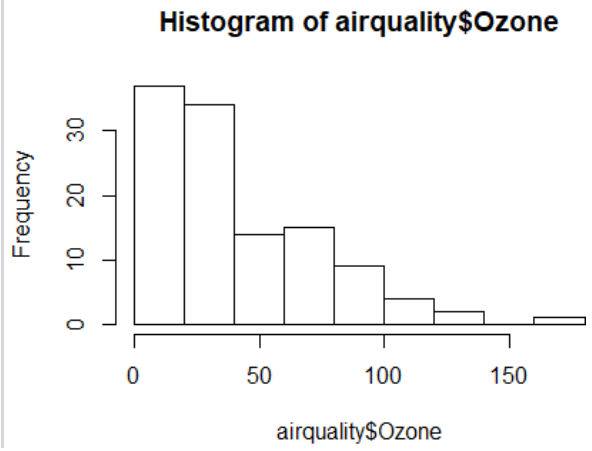
Base Graphics

Base graphics are used most commonly and are a very powerful system for creating 2-D graphics.

- There are two phases to creating a base plot
  - Initializing a new plot
  - Annotating (adding to) an existing plot
- Calling plot(x, y) or hist(x) will launch a graphics device (if one is not already open) and draw a new plot on the device.
- If the arguments to plot are not of some special class, then the default method for plot is called; this function has many arguments, letting you set the title, x axis label, y axis label, etc.
- The base graphics system has many parameters that can set and tweaked; these parameters are documented in ?par; it wouldn't hurt to try to memorize this help page!

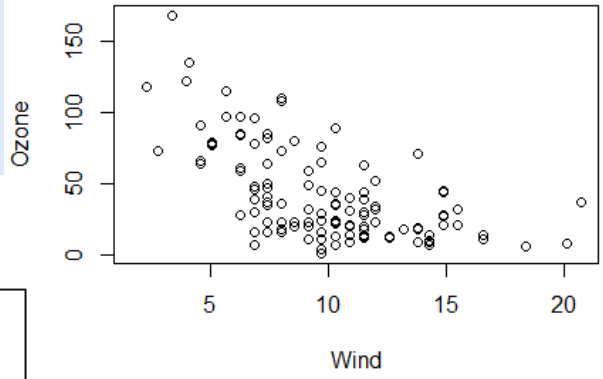
Simple Base Graphics

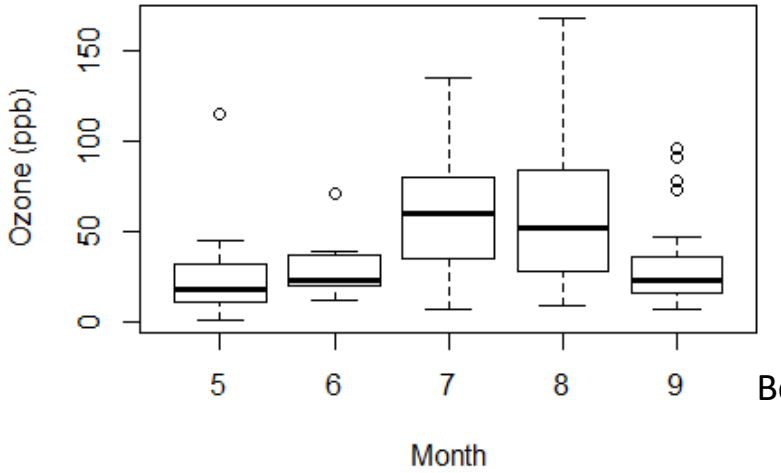
```
>
> library(datasets)
> hist(airquality$Ozone) ## Draw a new plot
>
```




```
>
> library(datasets)
> with(airquality, plot(Wind, Ozone))
>
```

Scatterplot of airquality





Boxplot of airquality

```
> library(datasets)
> airquality <- transform(airquality, Month = factor(Month))
> boxplot(Ozone ~ Month, airquality, xlab = "Month", ylab = "Ozone (ppb)")
```

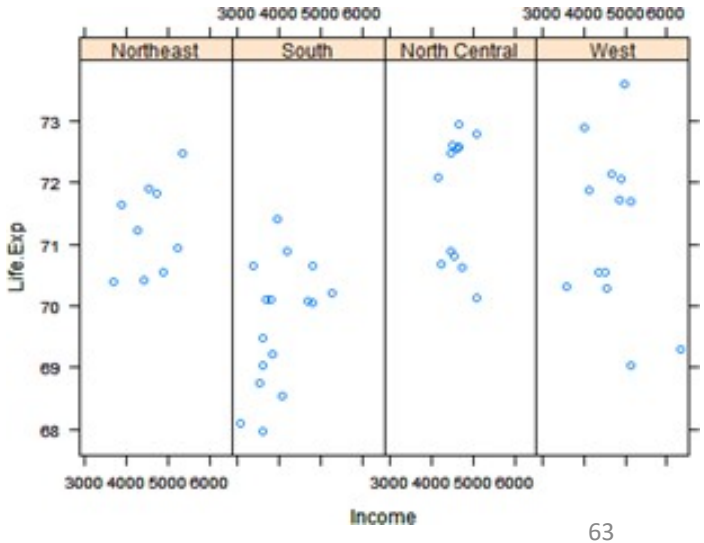
The Lattice System

- Plots are created with a single function call (xyplot, bwplot, etc.)
- Most useful for conditioning types of plots: Looking at how y changes with x across levels of z.
- Things like margins/spacing set automatically because entire plot is specified at once.
- Good for putting many plots on a screen.
- Sometimes awkward to specify an entire plot in a single function call.
- Annotation in plot is not especially intuitive.
- Use of panel functions and subscripts difficult to wield and requires intense preparation.
- Cannot "add" to the plot once it is created.

Lattice: Entire plot specified by one function; conditioning.

Lattice graphics are usually created in a single function call, so all of the graphics parameters have to specified at once; specifying everything at once allows R to automatically calculate the necessary spacing and font sizes.

```
>
> library(lattice)
> state <- data.frame(state.x77, region = state.region)
> xyplot(Life.Exp ~ Income | region, data = state, layout = c(4, 1))
>
```



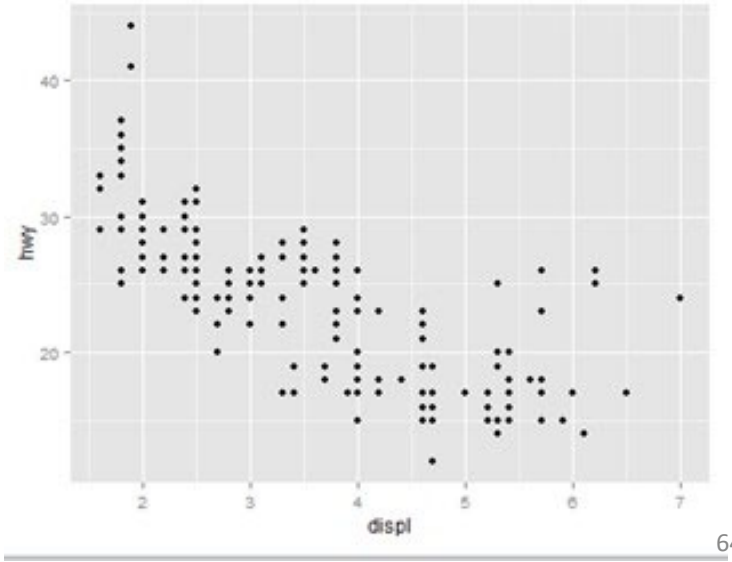
The ggplot2 System

- Splits the difference between base and lattice in a number of ways.
- Automatically deals with spacing, text, titles but also allows you to annotate by "adding" to a plot.
- Superficial similarity to lattice but generally easier/more intuitive to use.
- Default mode makes many choices for you (but you can still customize to your heart's desire).

ggplot2: Mixes elements of Base and Lattice

ggplot2 combines concepts from both base and lattice graphics but uses an independent implementation.

```
>
> library(ggplot2)
> data(mpg)
> qplot(displ, hwy, data = mpg)
>
```



Some Important Base Graphics Parameters

- Many base plotting functions share a set of parameters. Here are a few key ones:
  - pch: the plotting symbol (default is open circle)
  - lty: the line type (default is solid line), can be dashed, dotted, etc.
  - lwd: the line width, specified as an integer multiple
  - col: the plotting color, specified as a number, string, or hex code; the colors() function gives you a vector of colors by name
  - xlab: character string for the x-axis label
  - ylab: character string for the y-axis label

Default values for global graphics parameters

```
par("lty")
```

```
## [1] "solid"
```

```
par("col")
```

```
## [1] "black"
```

```
par("pch")
```

```
## [1] 1
```

Some Important Base Graphics Parameters

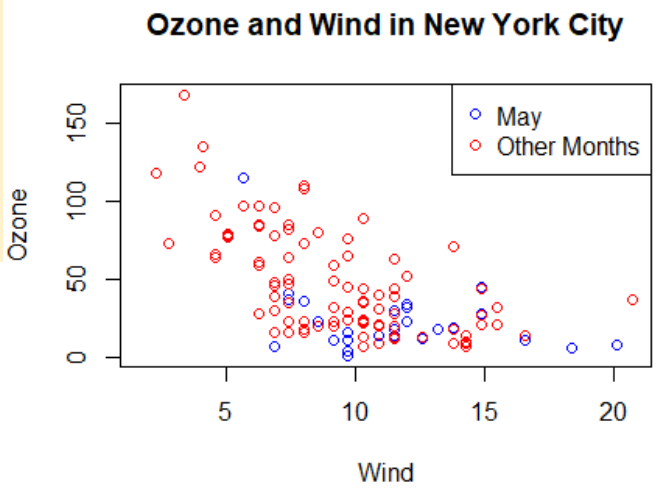
- The par() function is used to specify *global* graphics parameters that affect all plots in an R session. These parameters can be overridden when specified as arguments to specific plotting functions.
  - las: the orientation of the axis labels on the plot
  - bg: the background color
  - mar: the margin size
  - oma: the outer margin size (default is 0 for all sides)
  - mfrow: number of plots per row, column (plots are filled row-wise)
  - mfcol: number of plots per row, column (plots are filled column-wise)

Base Plotting Functions

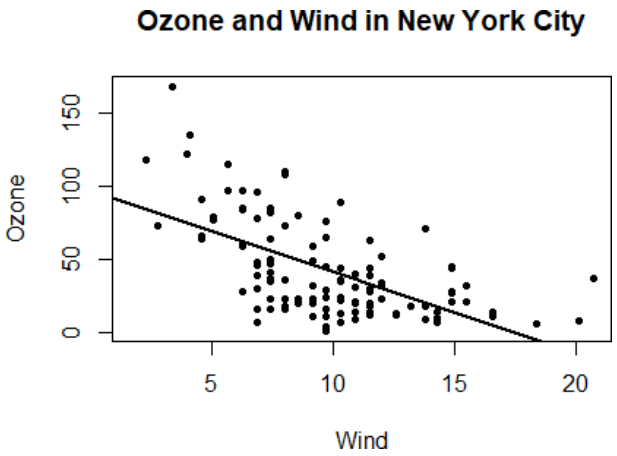
- **plot**: make a scatterplot, or other type of plot depending on the class of the object being plotted
- **lines**: add lines to a plot, given a vector x values and a corresponding vector of y values (or a 2- column matrix); this function just connects the dots.
- **points**: add points to a plot
- **text**: add text labels to a plot using specified x, y coordinates
- **title**: add annotations to x, y axis labels, title, subtitle, outer margin
- **mtext**: add arbitrary text to the margins (inner or outer) of the plot
- **axis**: adding axis ticks/labels

Base Plot with Annotation

```
> with(airquality, plot(Wind, Ozone, main = "Ozone and Wind in New York City",
+                       type = "n"))
> with(subset(airquality, Month == 5), points(Wind, Ozone, col = "blue"))
> with(subset(airquality, Month != 5), points(Wind, Ozone, col = "red"))
> legend("topright", pch = 1, col = c("blue", "red"), legend = c("May", "Other
Months"))
```

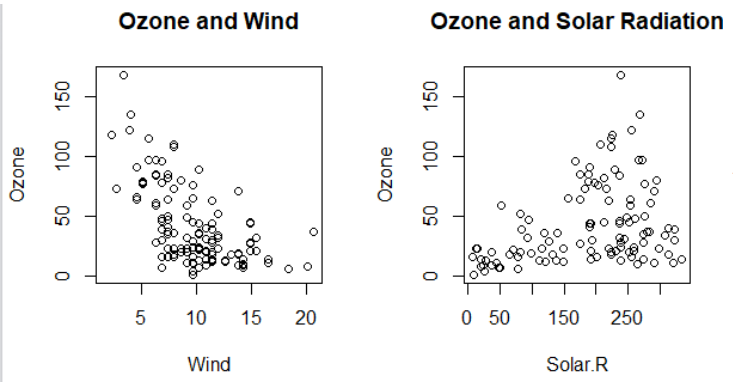


Base Plot with Regression Line



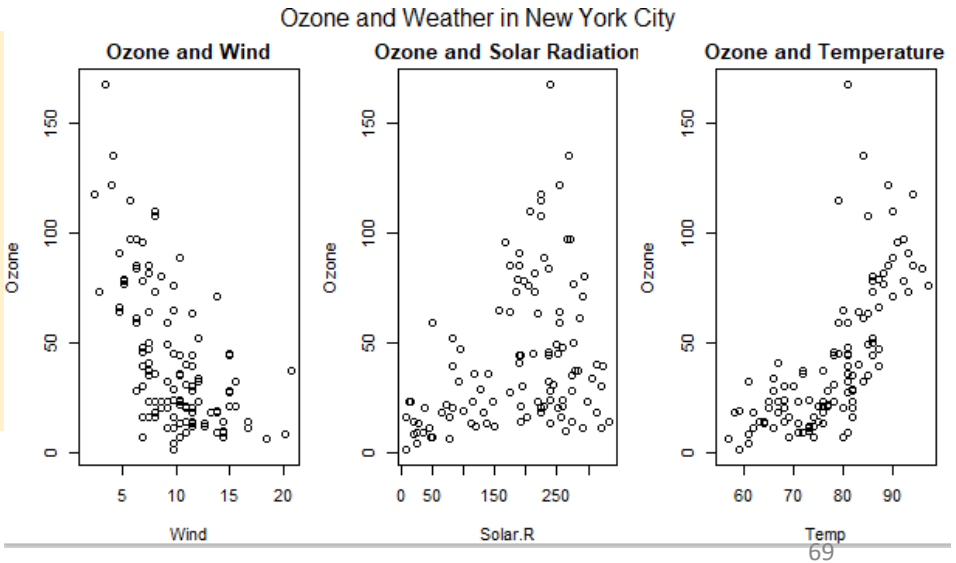

```
>with(airquality, plot(Wind, Ozone, main = "Ozone and Wind in
New York City",
+                       pch = 20))
> model <- lm(Ozone ~ Wind, airquality)
> abline(model, lwd = 2)
```

Multiple Base Plots



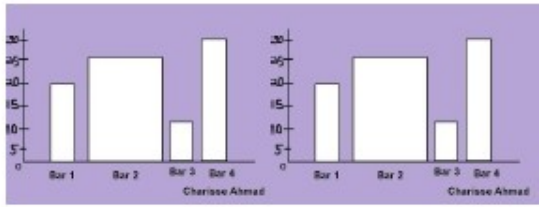

```
> par(mfrow = c(1, 2))
> with(airquality, {
+     plot(Wind, Ozone, main = "Ozone and Wind")
+     plot(Solar.R, Ozone, main = "Ozone and Solar Radiation")
+ })
```

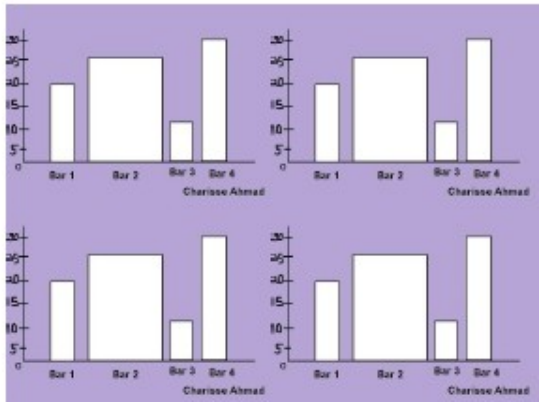
```
> par(mfrow = c(1, 3), mar = c(4, 4, 2, 1), oma = c(0, 0, 2, 0))
> with(airquality, {
+     plot(Wind, Ozone, main = "Ozone and Wind")
+     plot(Solar.R, Ozone, main = "Ozone and Solar Radiation")
+     plot(Temp, Ozone, main = "Ozone and Temperature")
+     mtext("Ozone and Weather in New York City", outer =
TRUE)
+ })
```



{base} graphics system – par()

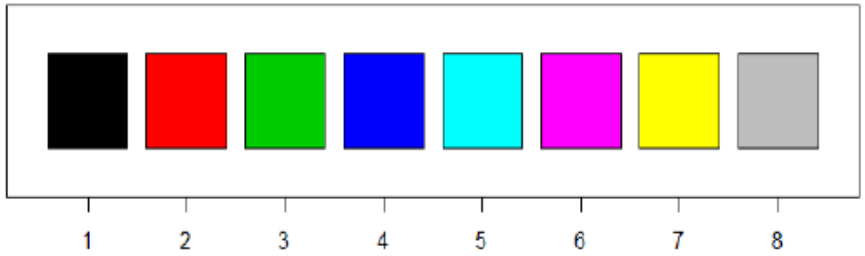
- Panel plotting
  - 1 by 2
  - 2 by 2
  - 3 by 1
- `par()`
  - `mfrows()`





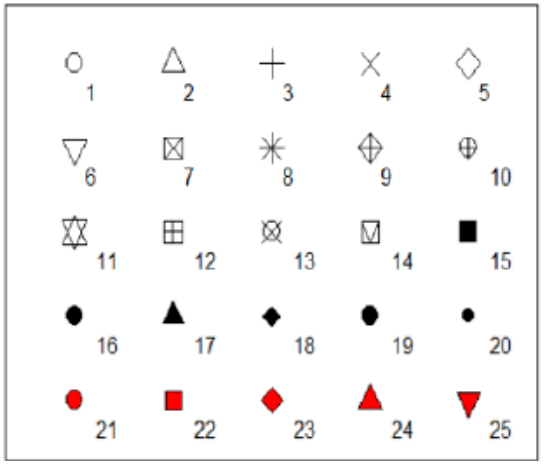
col and pch

col

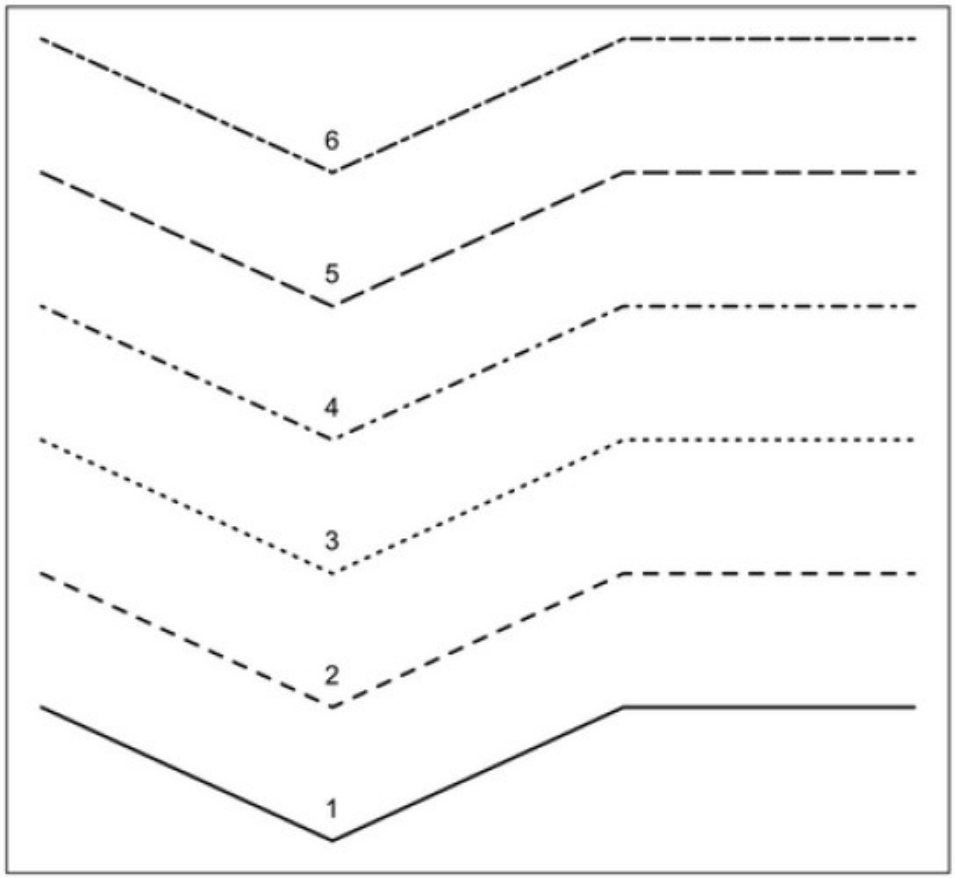


R colour chart:
http://research.stowers-institute.org/efg/R/Color/Chart/ColorChart.pdf

pch



Line types



Par () margins


```
par (mar=c(3,4,3,4), oma=c(3,4,3,4))
```

{base} graphics system – output

- Demonstration on the various common formats
  - PDF – `pdf()`
  - PNG – `png()`
  - JPG – `jpg()`
  - Screen (in MS-Windows systems, it's called `windows()`)
    - `dev.copy()`
    - `dev.copy2pdf()`

Graphics Device

A **graphics device** is something where you can make a plot appear.

- A window on your computer (screen device)
- A PDF file (file device)
- A PNG or JPEG file (file device)
- A scalable vector graphics (SVG) file (file device)

When you make a plot in R, it has to be "sent" to a specific **graphics device**.

The most common place for a plot to be "sent" is the screen device.

- On a Mac the screen device is launched with the quartz()
- On Windows the screen device is launched with windows()
- On Unix/Linux the screen device is launched with x11()

Reference: https://bookdown.org/rdpeng/exdata/graphics-devices.html

How Does a Plot Get Created?

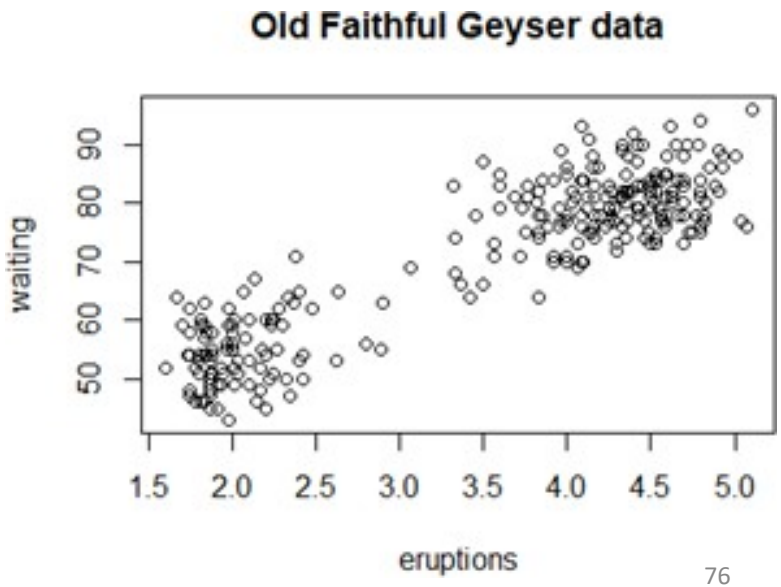
There are **two basic approaches** to plotting.

The **first approach** is most common:

1. Call a plotting function like plot, xyplot, or qplot
2. The plot appears on the screen device
3. Annotate plot if necessary
4. Try out

```
> library(datasets)
## Make plot appear on screen device
> with(faithful, plot(eruptions, waiting))

## Annotate with a title
> title(main = "Old Faithful Geyser data")
```



How Does a Plot Get Created?

The **second approach** to plotting is most commonly used for file devices:

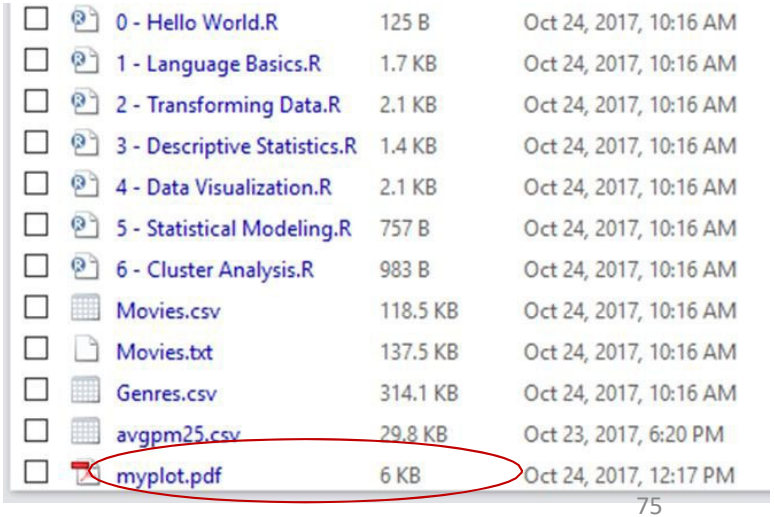
1. Explicitly launch a graphics device
2. Call a plotting function to make a plot (Note: if you are using a file device, no plot will appear on the screen)
3. Annotate plot if necessary
4. Explicitly close graphics device with dev.off() (this is very important!)

```
## Open PDF device; create 'myplot.pdf' in my working directory
> pdf(file = "myplot.pdf")

## Create plot and send to a file (no plot appears on screen)
> with(faithful, plot(eruptions, waiting))

## Annotate plot; still nothing on screen
> title(main = "Old Faithful Geyser data")

## Close the PDF file device
> dev.off()
## Now you can view the file 'myplot.pdf' on your computer
```



Graphics File Devices

There are two basic types of file devices: vector and bitmap devices

Vector formats:

- **pdf**: useful for line-type graphics, resizes well, usually portable, not efficient if a plot has many objects/points.
- **svg**: XML-based scalable vector graphics; supports animation and interactivity, potentially useful for web-based plots.
- **win.metafile**: Windows metafile format (only on Windows).
- **postscript**: older format, also resizes well, usually portable, can be used to create encapsulated postscript files; Windows systems often don't have a postscript viewer.

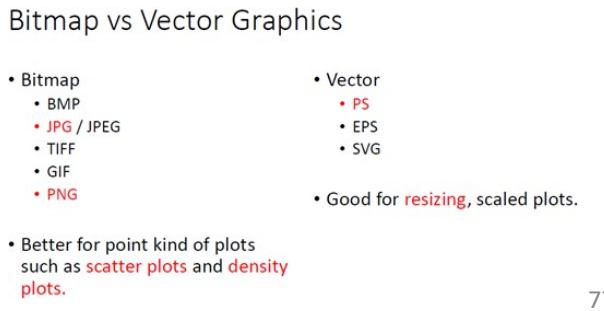
Vector formats are good for line drawings and plots with solid colors using a modest number of points.

Graphics File Devices

Bitmap formats:

- **png**: bitmapped format, good for line drawings or images with solid colors, uses lossless compression (like the old GIF format), most web browsers can read this format natively, good for plotting many points, does not resize well.
- **jpeg**: good for photographs or natural scenes, uses lossy compression, good for plotting many points, does not resize well, can be read by almost any computer and any web browser, not great for line drawings.
- **tiff**: Creates bitmap files in the TIFF format; supports lossless compression.
- **bmp**: a native Windows bitmapped format.

Bitmap formats are good for plots with a large number of points, natural scenes or web-based plots.



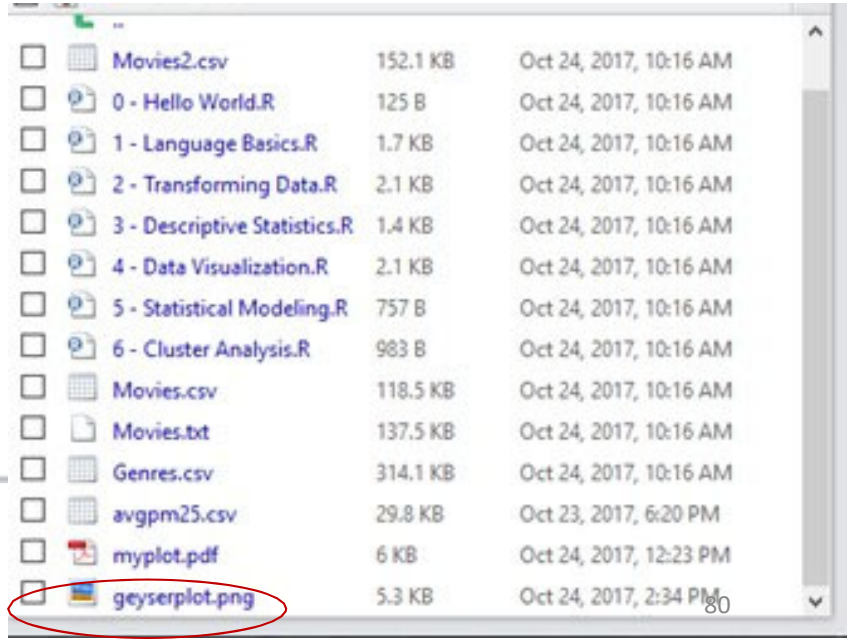
Copying Plots

Copying a plot to another device can be useful because some plots require a lot of code and it can be a pain to type all that in again for a different device.

- dev.copy: copy a plot from one device to another
- dev.copy2pdf: specifically copy a plot to a PDF file

NOTE: Copying a plot is not an exact operation, so the result may not be identical to the original.

```
>
> library(datasets)
> with(faithful, plot(eruptions, waiting)) ## Create plot on screen device
> title(main = "Old Faithful Geyser data") ## Add a main title
> dev.copy(png, file = "geyserplot.png") ## Copy my plot to a PNG file
png
  4
> dev.off() ## Don't forget to close the PNG device!
RStudioGD
        2
>
```



An Example - Let's Try Out

See how EDA is done in R.

1) Copy the file (**avgpm25.csv**) available on Spectrum into your work space.
    - Run the code individually and see (understand) the result.
    - Refer to slides 80 – 85.
2) Coursera EDA project example
    - Coursera EDA_project
    - M4Project2.R
    - summarySCC_PM25.rds
    - Source_Classification_Code.rds
3) For **SWIRL** – Refer to **Lesson 12: Looking at Data & Lesson 15: Base Graphics**

Air Pollution in the United States

- The U.S. Environmental Protection Agency (EPA) sets national ambient air quality standards for outdoor air pollution.
  - U.S. National Ambient Air Quality Standards
- For fine particle pollution (PM2.5), the "annual mean, averaged over 3 years" cannot exceed 12 $\mu g/m3$
- Data on daily PM2.5 are available from the U.S. EPA web site

  https://figshare.com/articles/Annual_Average_Pollution_in_the_US/1546752

  avgpm25.csv available on Spectrum
- **Question:** Are there any counties in the U.S. that exceed that national standard for fine particle pollution?

Reference: https://bookdown.org/rdpeng/exdata/exploratory-graphs.html

Data

Annual average PM2.5 averaged over the period 2008 through 2010.

```
>
>
> pollution<- read.csv("avgpm25.csv", colClasses = c("numeric", "character", "factor", "numeric", "numeric"
))
> head(pollution)
       pm25  fips region longitude latitude
1  9.771185 01003   east -87.74826 30.59278
2  9.993817 01027   east -85.84286 33.26581
3 10.688618 01033   east -87.72596 34.73148
4 11.337424 01049   east -85.79892 34.45913
5 12.119764 01055   east -86.03212 34.01860
6 10.827805 01069   east -85.35039 31.18973
>
```

Do any counties exceed the standard of 12μg/m^3?

Try this out, use the dataset “avgpm25.csv” available on Spectrum.

One Dimension

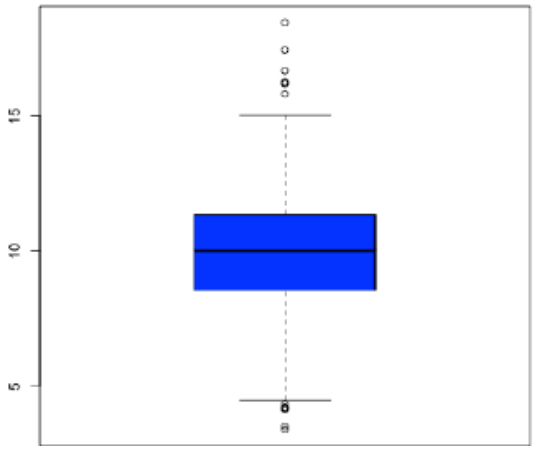
Five Number Summary

```
summary(pollution$pm25)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    3.38    8.55   10.00    9.84   11.40   18.40
```

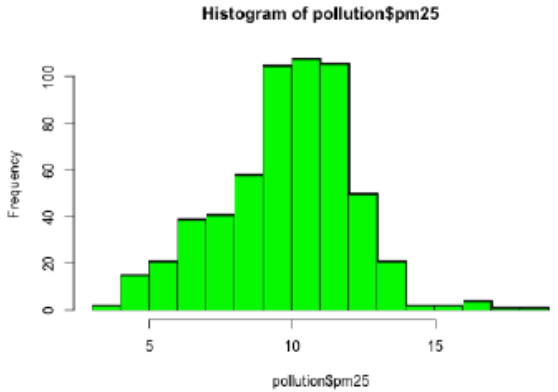
Boxplot

```
boxplot(pollution$pm25, col = "blue")
```

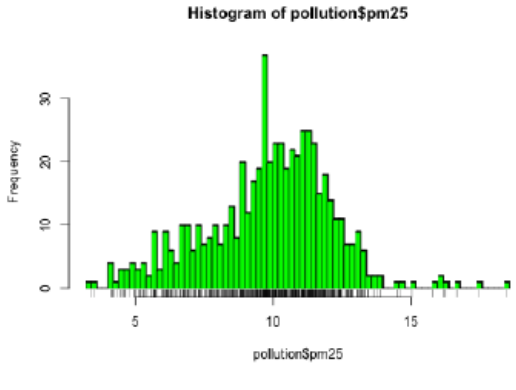


Histogram

```
hist(pollution$pm25, col = "green")
```



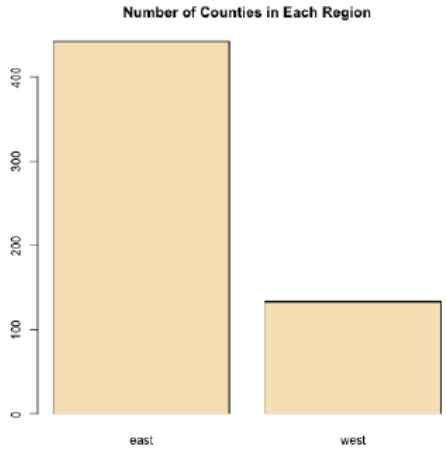

```
hist(pollution$pm25, col = "green", breaks = 100)
rug(pollution$pm25)
```



rug(data) = density plot, add a strip under the histogram indicating location of each data point

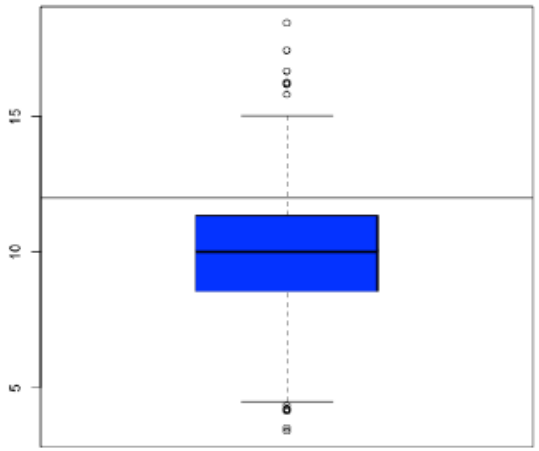
Barplot

```
barplot(table(pollution$region), col = "wheat", main = "Number of Counties in Each Region")
```

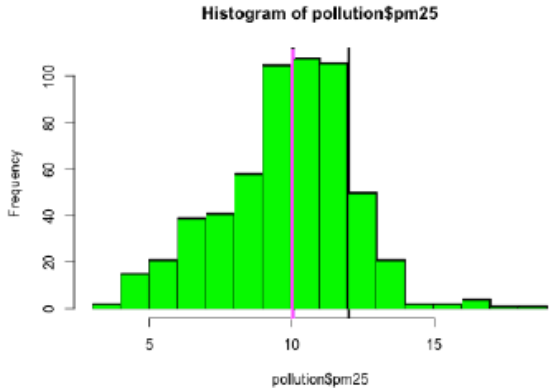


Overlaying Features

```
boxplot(pollution$pm25, col = "blue")
abline(h = 12)
```



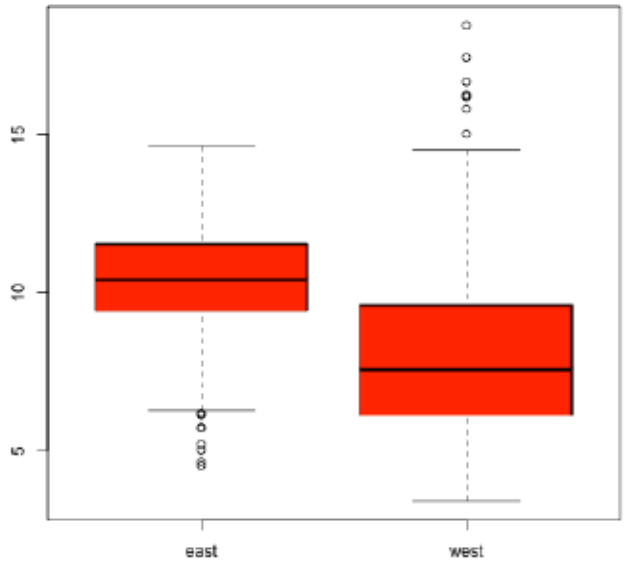

```
hist(pollution$pm25, col = "green")
abline(v = 12, lwd = 2)
abline(v = median(pollution$pm25), col = "magenta", lwd = 4)
```



Two Dimension

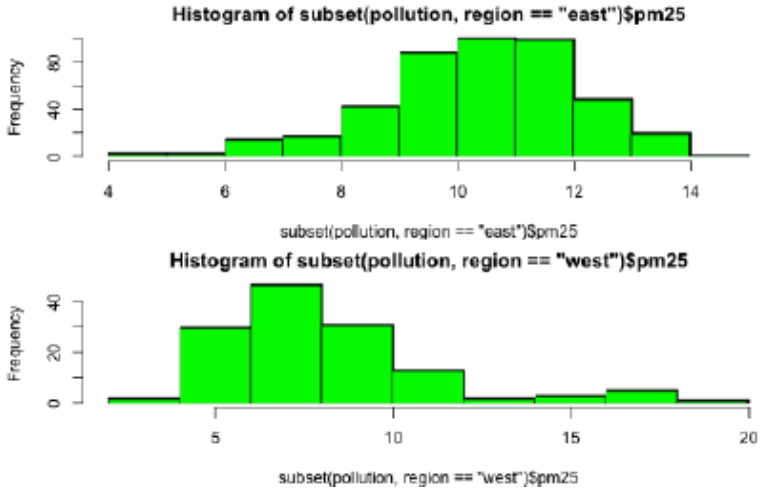
Multiple Boxplots

```
boxplot(pm25 ~ region, data = pollution, col = "red")
```



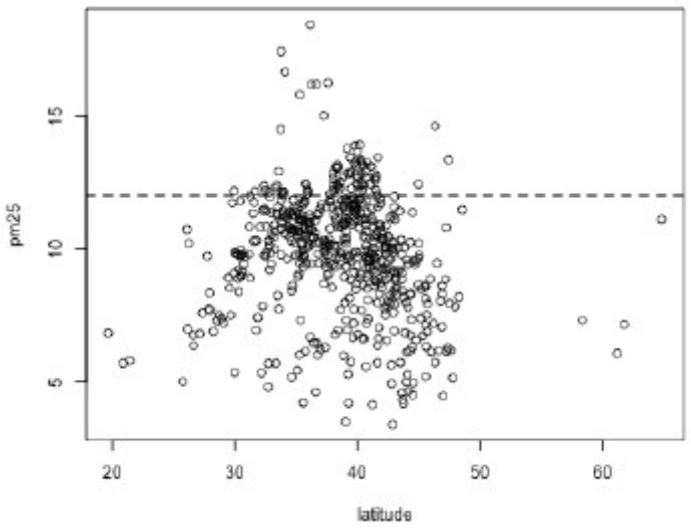
Multiple Histograms

```
par(mfrow = c(2, 1), mar = c(4, 4, 2, 1))
hist(subset(pollution, region == "east")$pm25, col = "green")
hist(subset(pollution, region == "west")$pm25, col = "green")
```



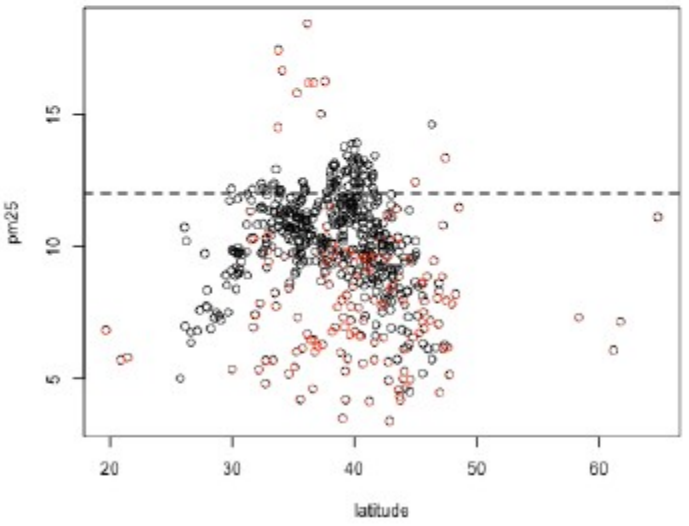
Scatterplot

```
with(pollution, plot(latitude, pm25))
abline(h = 12, lwd = 2, lty = 2)
```

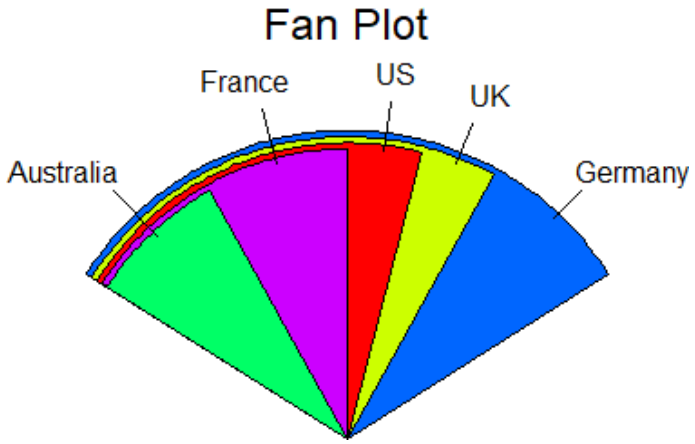


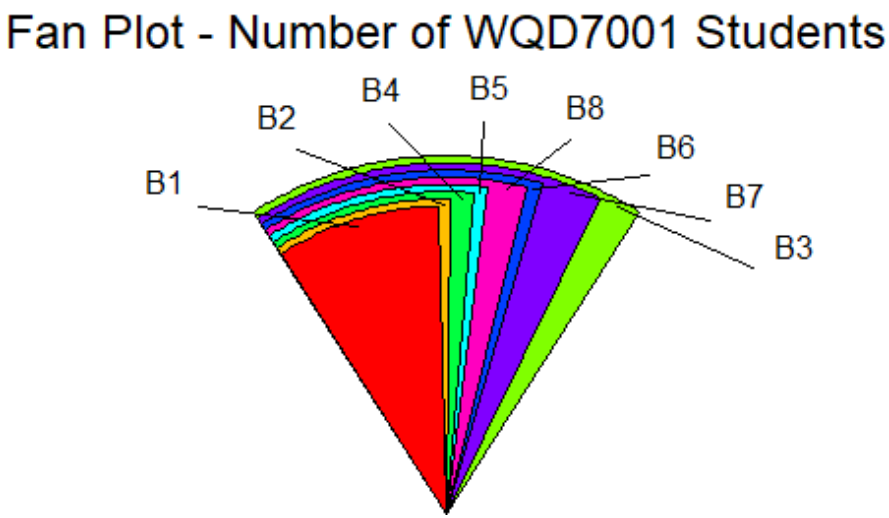
Scatterplot - Using Color

```
with(pollution, plot(latitude, pm25, col = region))
abline(h = 12, lwd = 2, lty = 2)
```



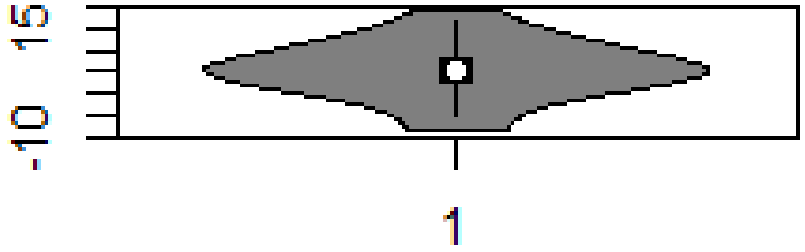
Fan Plot

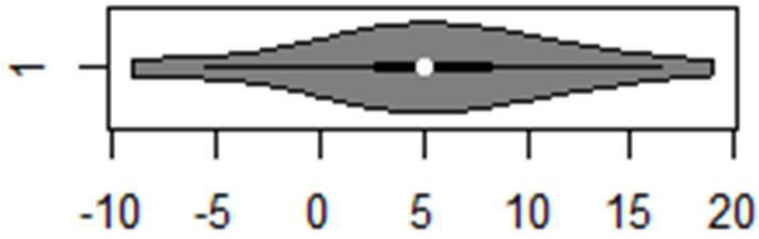





```
install.packages("plotrix")
library(plotrix)
slices <- c(38,41,80,46,49,60,72,57)
lbls <- c("B1", "B2", "B3", "B4", "B5", "B6", "B7", "B8")
fan.plot(slices, labels = lbls, main="Fan Plot - Number of WQD7001 Students")
```

Violin plot in R





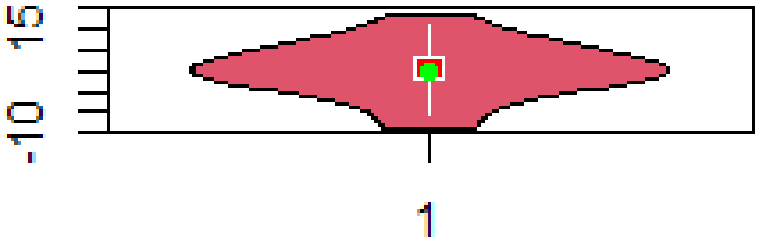

```
install.packages("vioplot")
library("vioplot")

x <- c(6, 9, 0, 19, -1, 8, 12, 5, 3, 7,
       2, 4, 3, -8, -9, 8, 4, 12, 5, 14)

vioplot(x)

vioplot(x, horizontal = TRUE)

vioplot(x,
        col = 2,                # Color of the area
        rectCol = "red",        # Color of the rectangle
        lineCol = "white",      # Color of the line
        colMed = "green",       # Pch symbol color
        border = "black",       # Color of the border of the violin
        pchMed = 16,            # Pch symbol for the median
        plotCentre = "points") # If "line", plots a median line
```



Source:- https://r-coder.com/violin-plot-r/

Top 50 ggplot2 Visualizations - The Master List (With Full R Code)

> What type of visualization to use for what sort of problem? This tutorial helps you choose the right type of chart for your specific objectives and how to implement it in R using ggplot2.

This is part 3 of a three part tutorial on ggplot2, an aesthetically pleasing (and very popular) graphics framework in R. This tutorial is primarily geared towards those having some basic knowledge of the R programming language and want to make complex and nice looking charts with R ggplot2.

- Part 1: Introduction to ggplot2, covers the basic knowledge about constructing simple ggplots and modifying the components and aesthetics.
- Part 2: Customizing the Look and Feel, is about more advanced customization like manipulating legend, annotations, multiplots with faceting and custom layouts

http://r-statistics.co/Top50-Ggplot2-Visualizations-MasterList-R-Code.html

Acknowledgments. These slides are adapted from the Coursera JHU Data Science Specialization course material

- R website: http://www.cran.r-project.org
- R Studio: http://www.rstudio.com
- Revolutions: http://blog.revolutionanalytics.com
- Flowing Data: http://flowingdata.com
- R-Blogger: http://www.r-bloggers.com

https://www.infoq.com/presentations/data-analysis-r

http://www.matthewrenze.com/workshops/practical-data-science-with-r/

Another Example of EDA - https://towardsdatascience.com/exploratory-data-analysis-8fc1cb20fd15