

Descriptive Statistics

Lecture 2

1 Outline

- Data Organization
- Measures of Central Tendency
- Measures of Dispersion
- Measure of Location
- Skewness and Kurtosis
- Basic Data Visualization using R

2 Data Organization

2.1 Data Exploration

In order to explore the underlying nature of the provided information, we need to explore the data. This exploration is made much easier if the data are organized and summarized.

i Raw Data

Raw data are measurements that have not been organized, summarized, or otherwise manipulated. As a biostatistician, it is common to be provided the raw data for any analysis.

2.2 Ordered Lists

The easiest step in organizing data is simply to order the data.

i Ordered Lists/Arrays

An ordered list is a list of values that are arranged from the smallest to largest value (or largest to smallest).

2.3 Ordered Lists in R

There are different ways to create an ordered list in R. Let us use the `iris` dataset in R and the `tidyverse()` package.

```
library(tidyverse)
```

```
-- Attaching core tidyverse packages ----- tidyverse 2.0.0 --
v dplyr      1.1.4      v readr      2.1.5
v forcats    1.0.0      v stringr    1.5.1
v ggplot2    3.5.2      v tibble     3.2.1
v lubridate  1.9.4      v tidyr      1.3.1
v purrr      1.0.4
-- Conflicts ----- tidyverse_conflicts() --
x dplyr::filter() masks stats::filter()
x dplyr::lag()     masks stats::lag()
i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become
```

```
glimpse(iris)
```

```
Rows: 150
Columns: 5
$ Sepal.Length <dbl> 5.1, 4.9, 4.7, 4.6, 5.0, 5.4, 4.6, 5.0, 4.4, 4.9, 5.4, 4.~
$ Sepal.Width  <dbl> 3.5, 3.0, 3.2, 3.1, 3.6, 3.9, 3.4, 3.4, 2.9, 3.1, 3.7, 3.~
$ Petal.Length <dbl> 1.4, 1.4, 1.3, 1.5, 1.4, 1.7, 1.4, 1.5, 1.4, 1.5, 1.5, 1.~
$ Petal.Width  <dbl> 0.2, 0.2, 0.2, 0.2, 0.2, 0.4, 0.3, 0.2, 0.2, 0.1, 0.2, 0.~
$ Species      <fct> setosa, setosa, setosa, setosa, setosa, setosa, setosa, setosa, s~
```

2.3.1 `sort()`

`sort(vector)` can arrange a vector in ascending or descending order. This works for both numeric and character vectors.

```
sort(iris$Sepal.Length)
```

```
[1] 4.3 4.4 4.4 4.4 4.5 4.6 4.6 4.6 4.6 4.7 4.7 4.8 4.8 4.8 4.8 4.8 4.9 4.9
[19] 4.9 4.9 4.9 4.9 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.1 5.1 5.1 5.1
[37] 5.1 5.1 5.1 5.1 5.1 5.2 5.2 5.2 5.2 5.3 5.4 5.4 5.4 5.4 5.4 5.4 5.5 5.5
[55] 5.5 5.5 5.5 5.5 5.5 5.6 5.6 5.6 5.6 5.6 5.6 5.7 5.7 5.7 5.7 5.7 5.7 5.7
[73] 5.7 5.8 5.8 5.8 5.8 5.8 5.8 5.8 5.9 5.9 5.9 6.0 6.0 6.0 6.0 6.0 6.0 6.1
[91] 6.1 6.1 6.1 6.1 6.1 6.2 6.2 6.2 6.2 6.3 6.3 6.3 6.3 6.3 6.3 6.3 6.3 6.3
[109] 6.4 6.4 6.4 6.4 6.4 6.4 6.4 6.5 6.5 6.5 6.5 6.5 6.6 6.6 6.7 6.7 6.7 6.7
[127] 6.7 6.7 6.7 6.7 6.8 6.8 6.8 6.9 6.9 6.9 6.9 7.0 7.1 7.2 7.2 7.2 7.3 7.4
[145] 7.6 7.7 7.7 7.7 7.7 7.9
```

```
sort(iris$Sepal.Length,decreasing = TRUE)
```

```
[1] 7.9 7.7 7.7 7.7 7.7 7.6 7.4 7.3 7.2 7.2 7.2 7.1 7.0 6.9 6.9 6.9 6.9 6.8
[19] 6.8 6.8 6.7 6.7 6.7 6.7 6.7 6.7 6.7 6.7 6.6 6.6 6.5 6.5 6.5 6.5 6.5 6.4
[37] 6.4 6.4 6.4 6.4 6.4 6.4 6.3 6.3 6.3 6.3 6.3 6.3 6.3 6.3 6.3 6.2 6.2 6.2
[55] 6.2 6.1 6.1 6.1 6.1 6.1 6.1 6.0 6.0 6.0 6.0 6.0 6.0 5.9 5.9 5.9 5.8 5.8
[73] 5.8 5.8 5.8 5.8 5.8 5.7 5.7 5.7 5.7 5.7 5.7 5.7 5.7 5.6 5.6 5.6 5.6 5.6
[91] 5.6 5.5 5.5 5.5 5.5 5.5 5.5 5.5 5.5 5.4 5.4 5.4 5.4 5.4 5.4 5.3 5.2 5.2
[109] 5.2 5.1 5.1 5.1 5.1 5.1 5.1 5.1 5.1 5.1 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0
[127] 5.0 5.0 4.9 4.9 4.9 4.9 4.9 4.9 4.8 4.8 4.8 4.8 4.8 4.7 4.7 4.6 4.6 4.6
[145] 4.6 4.5 4.4 4.4 4.4 4.3
```

2.3.2 arrange()

`arrange(dataframe,variable)` can sort a column in a data frame. This function is from the `dplyr` package, which is a part of `tidyverse`.

```
df <- arrange(iris,Sepal.Length)
glimpse(df)
```

Rows: 150

Columns: 5

```
$ Sepal.Length <dbl> 4.3, 4.4, 4.4, 4.4, 4.5, 4.6, 4.6, 4.6, 4.6, 4.7, 4.7, 4.~
$ Sepal.Width <dbl> 3.0, 2.9, 3.0, 3.2, 2.3, 3.1, 3.4, 3.6, 3.2, 3.2, 3.~
$ Petal.Length <dbl> 1.1, 1.4, 1.3, 1.3, 1.3, 1.5, 1.4, 1.0, 1.4, 1.3, 1.~
$ Petal.Width <dbl> 0.1, 0.2, 0.2, 0.2, 0.3, 0.2, 0.3, 0.2, 0.2, 0.2, 0.~
$ Species <fct> setosa, setosa, setosa, setosa, setosa, setosa, setosa, s~
```

```
df$Sepal.Length
```

```
[1] 4.3 4.4 4.4 4.4 4.5 4.6 4.6 4.6 4.6 4.7 4.7 4.8 4.8 4.8 4.8 4.8 4.9 4.9
[19] 4.9 4.9 4.9 4.9 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.1 5.1 5.1 5.1
[37] 5.1 5.1 5.1 5.1 5.1 5.2 5.2 5.2 5.2 5.3 5.4 5.4 5.4 5.4 5.4 5.4 5.5 5.5
[55] 5.5 5.5 5.5 5.5 5.5 5.6 5.6 5.6 5.6 5.6 5.6 5.7 5.7 5.7 5.7 5.7 5.7
[73] 5.7 5.8 5.8 5.8 5.8 5.8 5.8 5.8 5.9 5.9 5.9 6.0 6.0 6.0 6.0 6.0 6.1
[91] 6.1 6.1 6.1 6.1 6.1 6.2 6.2 6.2 6.2 6.3 6.3 6.3 6.3 6.3 6.3 6.3 6.3
[109] 6.4 6.4 6.4 6.4 6.4 6.4 6.4 6.5 6.5 6.5 6.5 6.5 6.6 6.6 6.7 6.7 6.7
[127] 6.7 6.7 6.7 6.7 6.8 6.8 6.8 6.9 6.9 6.9 6.9 7.0 7.1 7.2 7.2 7.2 7.3
[145] 7.6 7.7 7.7 7.7 7.7 7.9
```

```
df <- arrange(iris,desc(Sepal.Length))
glimpse(df)
```

Rows: 150

Columns: 5

```
$ Sepal.Length <dbl> 7.9, 7.7, 7.7, 7.7, 7.7, 7.6, 7.4, 7.3, 7.2, 7.2, 7.2, 7.~
$ Sepal.Width <dbl> 3.8, 3.8, 2.6, 2.8, 3.0, 3.0, 2.8, 2.9, 3.6, 3.2, 3.0, 3.~
$ Petal.Length <dbl> 6.4, 6.7, 6.9, 6.7, 6.1, 6.6, 6.1, 6.3, 6.1, 6.0, 5.8, 5.~
$ Petal.Width <dbl> 2.0, 2.2, 2.3, 2.0, 2.3, 2.1, 1.9, 1.8, 2.5, 1.8, 1.6, 2.~
$ Species <fct> virginica, virginica, virginica, virginica, virginica, vi~
```

```
df$Sepal.Length
```

```
[1] 7.9 7.7 7.7 7.7 7.7 7.6 7.4 7.3 7.2 7.2 7.2 7.1 7.0 6.9 6.9 6.9 6.9 6.8
[19] 6.8 6.8 6.7 6.7 6.7 6.7 6.7 6.7 6.7 6.7 6.6 6.6 6.5 6.5 6.5 6.5 6.5 6.4
[37] 6.4 6.4 6.4 6.4 6.4 6.4 6.3 6.3 6.3 6.3 6.3 6.3 6.3 6.3 6.3 6.2 6.2 6.2
[55] 6.2 6.1 6.1 6.1 6.1 6.1 6.1 6.0 6.0 6.0 6.0 6.0 6.0 5.9 5.9 5.9 5.8 5.8
[73] 5.8 5.8 5.8 5.8 5.8 5.7 5.7 5.7 5.7 5.7 5.7 5.7 5.7 5.6 5.6 5.6 5.6 5.6
[91] 5.6 5.5 5.5 5.5 5.5 5.5 5.5 5.5 5.5 5.4 5.4 5.4 5.4 5.4 5.4 5.3 5.2 5.2
[109] 5.2 5.1 5.1 5.1 5.1 5.1 5.1 5.1 5.1 5.1 5.1 5.0 5.0 5.0 5.0 5.0 5.0 5.0
[127] 5.0 5.0 4.9 4.9 4.9 4.9 4.9 4.9 4.8 4.8 4.8 4.8 4.8 4.8 4.7 4.7 4.6 4.6
[145] 4.6 4.5 4.4 4.4 4.4 4.3
```

2.4 Ordered Lists: Pros and Cons

The ordered list makes it easier to see the data sorted by a specific variable. However, it may be impractical to use for large data sets.

2.5 Exercise

The data set `USArrests` contains statistics, in arrests per 100,000 residents for assault, murder, and rape in each of the 50 US states in 1973.

2.5.1 Exercise

Create an ordered list for the `Assault` variable in `USArrests`. Which state had the highest rate for assault per 100,000 residents?

2.5.2 Answer

```
sort(USArrests$Assault,decreasing = T)
```

```
[1] 337 335 300 294 285 279 276 263 259 255 254 252 249 249 238 236 211 204 201
[20] 190 188 178 174 161 159 159 156 151 149 145 120 120 120 115 113 110 109 109
[39] 106 102 86 83 81 72 57 56 53 48 46 45
```

```
arrange(USArrests,desc(Assault))
```

	Murder	Assault	UrbanPop	Rape
North Carolina	13.0	337	45	16.1
Florida	15.4	335	80	31.9
Maryland	11.3	300	67	27.8
Arizona	8.1	294	80	31.0
New Mexico	11.4	285	70	32.1
South Carolina	14.4	279	48	22.5
California	9.0	276	91	40.6
Alaska	10.0	263	48	44.5
Mississippi	16.1	259	44	17.1
Michigan	12.1	255	74	35.1
New York	11.1	254	86	26.1
Nevada	12.2	252	81	46.0
Illinois	10.4	249	83	24.0
Louisiana	15.4	249	66	22.2
Delaware	5.9	238	72	15.8
Alabama	13.2	236	58	21.2
Georgia	17.4	211	60	25.8
Colorado	7.9	204	78	38.7

Texas	12.7	201	80	25.5
Arkansas	8.8	190	50	19.5
Tennessee	13.2	188	59	26.9
Missouri	9.0	178	70	28.2
Rhode Island	3.4	174	87	8.3
Wyoming	6.8	161	60	15.6
New Jersey	7.4	159	89	18.8
Oregon	4.9	159	67	29.3
Virginia	8.5	156	63	20.7
Oklahoma	6.6	151	68	20.0
Massachusetts	4.4	149	85	16.3
Washington	4.0	145	73	26.2
Idaho	2.6	120	54	14.2
Ohio	7.3	120	75	21.4
Utah	3.2	120	80	22.9
Kansas	6.0	115	66	18.0
Indiana	7.2	113	65	21.0
Connecticut	3.3	110	77	11.1
Kentucky	9.7	109	52	16.3
Montana	6.0	109	53	16.4
Pennsylvania	6.3	106	72	14.9
Nebraska	4.3	102	62	16.5
South Dakota	3.8	86	45	12.8
Maine	2.1	83	51	7.8
West Virginia	5.7	81	39	9.3
Minnesota	2.7	72	66	14.9
New Hampshire	2.1	57	56	9.5
Iowa	2.2	56	57	11.3
Wisconsin	2.6	53	66	10.8
Vermont	2.2	48	32	11.2
Hawaii	5.3	46	83	20.2
North Dakota	0.8	45	44	7.3

North Carolina had the highest rate of Assault arrests per 100,000 residents with 337.

2.6 Frequency Tables: Categorical Variables

One way to summarize a data set is through frequency tables, which counts the frequency of occurrence of values in the data set. To group a set of observations, we select a set of contiguous, nonoverlapping intervals such that each value in the set of observations can be placed in one, and only one, of the intervals.

Note

For categorical variables, these intervals can be defined by the categories in the variable.

Important

Frequency tables are also referred to as frequency distributions.

2.7 Frequency Tables: Categorical Variables

Here is an example of a frequency table. The data set `flights` in the `nycflights23` package includes data for all flights that departed from airports in New York city. Recall that this data set has 435,452 rows. If we wished to determine where most flights originated from (column `origin`), we can count the number of rows that mentioned each of the airports.

To create a basic frequency table, we can use the function `count(dataframe,variable)` in the `dplyr` package.

```
library(nycflights23)
```

Warning: package 'nycflights23' was built under R version 4.5.1

```
count(flights,origin)
```

```
# A tibble: 3 x 2
  origin      n
  <chr>   <int>
1 EWR    138578
2 JFK    133048
3 LGA    163726
```

2.8 Exercise

For the following exercise, use the `infert` data set, which has data on education level, age, parity (previous pregnancies), and the incidence of infertility after spontaneous and induced abortion.

2.8.1 Exercise

After examining the data set using `glimpse`, create a frequency table for the education level of the subjects given by the variable `education`.

2.8.2 Answer

```
count(infert, education)
```

	education	n
1	0-5yrs	12
2	6-11yrs	120
3	12+ yrs	116

2.9 Relative Frequency

It may be useful at times to know the proportion, rather than the number, of values falling within a particular class interval. We obtain this information by dividing the number of values in the particular class interval by the total number of values. For an interval with k occurrences out of a total of N events, the relative frequency can be calculated by:

$$Rel.Freq. = n/N$$

Note

The relative frequency can be reported as a decimal or as a percentage ($\times 100\%$).

2.10 Cumulative Relative Frequency

We may sum, or cumulate, the relative frequencies to facilitate obtaining information regarding the relative frequency of values within two or more contiguous class intervals.

Note

The sum of *relative* frequencies is referred to as the **cumulative relative frequency**

origin	n
EWR	138578
JFK	133048
LGA	163726

2.11 Calculating Relative and Cumulative Frequencies

Let's try to calculate these frequencies by hand using the `origin` frequency distribution example from `nycflights`.

2.11.1 Relative Freq.

For EWR: $\frac{138578}{(138578+133048+163726)} = 0.3183125$

For JFK: $\frac{133048}{(138578+133048+163726)} = 0.3056102$

For LGA: $\frac{163726}{(138578+133048+163726)} = 0.3760773$

2.11.2 Cumulative Freq.

For EWR: 138578

For EWR + JFK: $138578 + 133048 = 271626$

For EWR+JFK + LGA: $(138578 + 133048 + 163726) = 435352$

2.11.3 Cumulative Rel. Freq.

For EWR: $\frac{138578}{(138578+133048+163726)} = 0.3183125$

For EWR + JFK: $\frac{138578+133048}{(138578+133048+163726)} = 0.6239227$

For EWR+JFK + LGA: $\frac{133048+138578+163726}{(138578+133048+163726)} = 1$

2.12 Frequency Tables in R

There are a lot of ways to create frequency tables in R. One of the easiest and most complete ways is to use the function `freq` in the package `summarytools`.

```
# install.packages("summarytools") # if you have not installed summarytools yet.
library(summarytools)
```

Warning: package 'summarytools' was built under R version 4.5.1

Attaching package: 'summarytools'

The following object is masked from 'package:tibble':

view

```
freq(flights$origin)
```

Frequencies
flights\$origin
Type: Character

	Freq	% Valid	% Valid Cum.	% Total	% Total Cum.
EWR	138578	31.83	31.83	31.83	31.83
JFK	133048	30.56	62.39	30.56	62.39
LGA	163726	37.61	100.00	37.61	100.00
<NA>	0			0.00	100.00
Total	435352	100.00	100.00	100.00	100.00

! Important

The <NA> row also counts rows that have missing values. The %Valid column only counts non-missing data, while the % Total column counts non-missing data in calculating relative and cumulative frequencies. In this example, there are no missing rows, hence the columns are the same.

The function also does not output cumulative frequencies, but usually cumulative relative frequencies suffice.

2.13 Frequency Tables: Quantitative Variables

These intervals are usually referred to as **class intervals**.

! Important

Too few intervals are undesirable because of the resulting loss of information. On the other hand, if too many intervals are used, the objective of summarization will not be met.

A commonly followed rule of thumb states that there should be no fewer than 5 intervals and no more than 15. However, if the number of intervals fall outside this range, it should be based on theoretical/foundational concepts that could justify the choice of intervals.

2.14 Frequency Tables in R: QV

Like categorical variables, there are a lot of ways to create a frequency tables for quantitative variables. However, using `freq()` directly on a quantitative variable could lead to trivial frequency tables.

⚠ Warning

What happens if you run the following code?

```
freq(iris$Sepal.Length)
```

2.15 Frequency Tables in R: hist()

2.15.1 hist(vector)

The function `hist()` creates both a histogram or a vector of frequencies. To show the vector of frequencies, the plotting mechanism must be turned off. This can be done by setting `plot=F`. For example,

```
hist(iris$Sepal.Length,plot=F)
```

```
$breaks
```

```
[1] 4.0 4.5 5.0 5.5 6.0 6.5 7.0 7.5 8.0
```

```
$counts
```

```
[1] 5 27 27 30 31 18 6 6
```

```
$density
```

```
[1] 0.06666667 0.36000000 0.36000000 0.40000000 0.41333333 0.24000000 0.08000000
```

```
[8] 0.08000000

$mids
[1] 4.25 4.75 5.25 5.75 6.25 6.75 7.25 7.75

$xname
[1] "iris$Sepal.Length"

$equidist
[1] TRUE

attr(,"class")
[1] "histogram"
```

i Note

In this output:

- **Breaks** are the endpoints of the intervals used to create the frequency table.
- **counts** are the frequencies for each interval.
- **mids** are the midpoints of each interval, which is commonly used as the representative numbers of each interval.

2.16 Frequency Tables in R: `cut(vector,breaks)` and `freq(vector)`

`cut()` creates a factor including the intervals for the quantitative variable based on the breaks you provide it. You can provide the breaks using the `seq()` function or the `c()` function.

```
# We must first create a vector of intervals for our frequency table.
ranges <- cut(iris$Sepal.Length,breaks=seq(from=4,to=8,by=0.5)) ①
head(ranges) ②
```

- ① `seq(from=4,to=8,by=0.5)` provides a vector of numbers from 4 to 8 in increments of 0.5.
- ② The “(x,y)” notation in the output means that the interval includes y, but not x. The `head()` function only shows the first 6 elements of `ranges`.

```
[1] (5,5.5] (4.5,5] (4.5,5] (4.5,5] (4.5,5] (5,5.5]
Levels: (4,4.5] (4.5,5] (5,5.5] (5.5,6] (6,6.5] (6.5,7] (7,7.5] (7.5,8]
```

We can now make a frequency table out of the intervals provided by `cut()`.

```
freq(ranges)
```

```
Frequencies  
ranges  
Type: Factor
```

	Freq	% Valid	% Valid Cum.	% Total	% Total Cum.
(4,4.5]	5	3.33	3.33	3.33	3.33
(4.5,5]	27	18.00	21.33	18.00	21.33
(5,5.5]	27	18.00	39.33	18.00	39.33
(5.5,6]	30	20.00	59.33	20.00	59.33
(6,6.5]	31	20.67	80.00	20.67	80.00
(6.5,7]	18	12.00	92.00	12.00	92.00
(7,7.5]	6	4.00	96.00	4.00	96.00
(7.5,8]	6	4.00	100.00	4.00	100.00
<NA>	0			0.00	100.00
Total	150	100.00	100.00	100.00	100.00

2.17 Exercise

Use the `infert` data set, which has data on education level, age, parity (previous pregnancies), and the incidence of infertility after spontaneous and induced abortion.

2.17.1 Exercise

Create a frequency distribution table with relative and cumulative relative frequencies for the education level of the subjects given by the variable `education`.

2.17.2 Answer

```
freq(infert$education)
```

```
Frequencies  
infert$education  
Type: Factor
```

	Freq	% Valid	% Valid Cum.	% Total	% Total Cum.
--	------	---------	--------------	---------	--------------

0-5yrs	12	4.84	4.84	4.84	4.84
6-11yrs	120	48.39	53.23	48.39	53.23
12+ yrs	116	46.77	100.00	46.77	100.00
<NA>	0			0.00	100.00
Total	248	100.00	100.00	100.00	100.00

3 Measures of Central Tendency

3.1 Descriptive Measures

Although frequency distributions serve useful purposes, there are many situations that require other types of data summarization. What we need in many instances is the ability to summarize the data by means of a single number called a **descriptive measure**. Descriptive measures may be computed from the data of a sample or the data of a population.

Statistic vs. Parameter

Statistics are descriptive measures computed from a sample. Statistics are typically represented by standard alphabet symbols (\bar{x}, p, r).

Parameters are descriptive measures computed from a population. Parameters are typically represented by Greek letters (μ, π, ρ).

3.2 Measures of Central Tendency

Measures of central tendency provides information on where the central point of the data is. The most common measures of central tendency are:

- Mean
- Median
- Mode

3.3 Mean

The mean, specifically the *arithmetic mean*, is the most familiar measure of central tendency.

3.3.1 Population Mean

i Population Mean

The population mean is calculated by adding all the values in the population and dividing by the population size N . Formally, for a vector of values given by $(X_1, X_2, X_3, \dots, X_N)$ defining the population, the mean μ can be calculated using the following equation:

$$\mu = \frac{(X_1 + X_2 + X_3 + \dots + X_N)}{N} = \frac{\sum_{i=1}^N X_i}{N}$$

3.3.2 Sample Mean

i Sample Mean

The sample mean is calculated by adding all the values in the sample and dividing by the sample size k . Formally, for a vector of values given by $(X_1, X_2, X_3, \dots, X_k)$ defining the population, the mean \bar{x} can be calculated using the following equation:

$$\bar{x} = \frac{(X_1 + X_2 + X_3 + \dots + X_k)}{k} = \frac{\sum_{i=1}^k X_i}{k}$$

3.4 Mean: Example

What is the mean of the following numbers: 5, 27, 26, 30, 31?

3.4.1 Math Method

The mean can be calculated using the formula $(5 + 27 + 26 + 30 + 31)/5 = 23.8$

3.4.2 R method

The function `mean()` outputs the mean of a vector of numbers. Remember to always put the numbers inside the `c()` function to form a vector.

```
mean(c(5,27,26,30,31))
```

```
[1] 23.8
```

3.5 Mean: Example

The function `mean()` can also be used to calculate means of quantitative variables in data frames.

```
mean(iris$Sepal.Length)
```

```
[1] 5.843333
```

3.6 Mean: Exercise

The data set `cars` include the speed (`speed`, in mph) and stopping distance (`dist`, in ft) of 50 cars recorded in the 1920s.

3.6.1 Exercise

Find the mean stopping distance for these 50 cars.

3.6.2 Answer

```
mean(cars$dist)
```

```
[1] 42.98
```

The mean stopping distance is 42.98 ft.

3.7 Mean: Advantages and Disadvantages

i Advantages

The mean is unique to a specific set of values, i.e. there is only one mean for every data set. The mean is also relatively easy to calculate and is a well-known summary statistic.

i Disadvantages

The mean is easily influenced by extreme values. As an example, recall the mean of the following numbers:


```
mean(c(5,27,26,30,31))
```

```
[1] 23.8
```

Now, if 31 was replaced by an extreme number, say 310, the new mean would be very different.

```
mean(c(5,27,26,30,310))
```

```
[1] 79.6
```

3.8 Median

The median is the middle value of ordered data.

! Important

If there is an odd number of observations, the median is a value in the middle of the data.

If there is an even number of observations, the median is the average of the two middle values.

3.9 Median: Example

3.9.1 Example 1: Odd

What is the median of the following numbers? 83.07 72.15 89.61 81.68 87.26

- Sort the values first in ascending order: 72.15, 81.68, 83.07, 87.26, 89.61
- There are five values, which means there is a middle value (83.07). Hence, the median is **83.07**.

3.9.2 Example 2: Even

What is the median of the following numbers? 100 3 7 5 8 2

- Sort the values first in ascending order: 2 3 5 7 8 100
- There are six values, which means there is no middle value. We need to take the average of the two middle values (5,7). Hence, the median is $(5+7)/2 = 6$.

3.10 Median: R

In R, we can use the `median(vector)` function.

```
median(c(83.07, 72.15, 89.61, 81.68, 87.26))
```

```
[1] 83.07
```

```
median(c(100, 3, 7, 5, 8, 2))
```

```
[1] 6
```

This works for variables in data frames.

```
median(iris$Sepal.Length)
```

```
[1] 5.8
```

3.11 Median: Exercise in R.

3.12 Exercise

The data set `USArrests` contains statistics, in arrests per 100,000 residents for assault, murder, and rape in each of the 50 US states in 1973.

3.12.1 Exercise

Create an ordered list for the `Assault` variable in `USArrests`. What is the median rate for assault per 100,000 residents?

3.12.2 Answer

```
median(USArrests$Assault)
```

```
[1] 159
```

The median arrest rate per 100,000 residents in the US in 1973 was 159.

3.13 Median: Advantages and Disadvantages

Advantages

The median is unique to a specific set of values, i.e. there is only one median for every data set. The median is also relatively easy to calculate once the data is ordered. Unlike the mean, the median is **NOT** influenced by extreme values.

```
median(c(5,27,26,30,31))
```

```
[1] 27
```

Now, if 31 was replaced by an extreme number, say 310, the new median would be the same because the middle part of the data did not change.

```
median(c(5,27,26,30,310))
```

```
[1] 27
```

Disadvantages

While the median is a well-known summary statistic, it is limited in the area of inferential statistics. The mean is more versatile as an estimator when it comes to statistical tests compared to the median.

3.14 Mode

For a sample of quantitative/qualitative data, the mode is the value that occurs most frequently.

Important

If all values occurred with the same frequency, then the data does not have a mode.

3.15 Mode: Example

What is the mode of the following values? 33, 35, 35, 46, 21, 56, 390.

- The mode is 35 because it occurred the most times.

What is the mode of the following values? 21, 35, 35, 46, 21, 56, 390.

- The modes are 21 and 35 because these values occurred twice, which was the highest frequency across the values.

What is the mode of the following values? 33, 34, 35, 46, 21, 56, 390.

- There is **NO** mode because all values occurred only once.

3.16 Mode: R

We can use the `freq()` or `hist()` functions in R to create a frequency table for the data set, but we often only need the frequency distribution. One function that we can use is the `table` function.

```
freq_table <- table(iris$Sepal.Length)
sort(freq_table,decreasing = TRUE)
```

```

 5 5.1 6.3 5.7 6.7 5.5 5.8 6.4 4.9 5.4 5.6  6 6.1 4.8 6.5 4.6 5.2 6.2 6.9 7.7
10  9  9  8  8  7  7  7  6  6  6  6  6  5  5  4  4  4  4  4
4.4 5.9 6.8 7.2 4.7 6.6 4.3 4.5 5.3  7 7.1 7.3 7.4 7.6 7.9
 3  3  3  3  2  2  1  1  1  1  1  1  1  1  1  1
```

```
freq_table <- table(iris$Species)
sort(freq_table,decreasing = TRUE)
```

```

setosa versicolor virginica
    50         50         50
```

The mode sepal length in the `iris` data set is 5. There is no mode for the species variable as all species occurred 50 times.

3.17 Mode: Advantages and Disadvantages

i Advantages

The mode is easily calculable using frequency tables. Unlike the mean and median, it can be calculated for qualitative variables.

i Disadvantages

The mode is not unique as data sets can have more than one mode. Modes are not used in inferential statistics for quantitative variables.

4 Measures of Dispersion

4.1 Dispersion

The dispersion of a set of observations refers to the variability they exhibit. Measures of dispersion convey information on the spread of a data set.

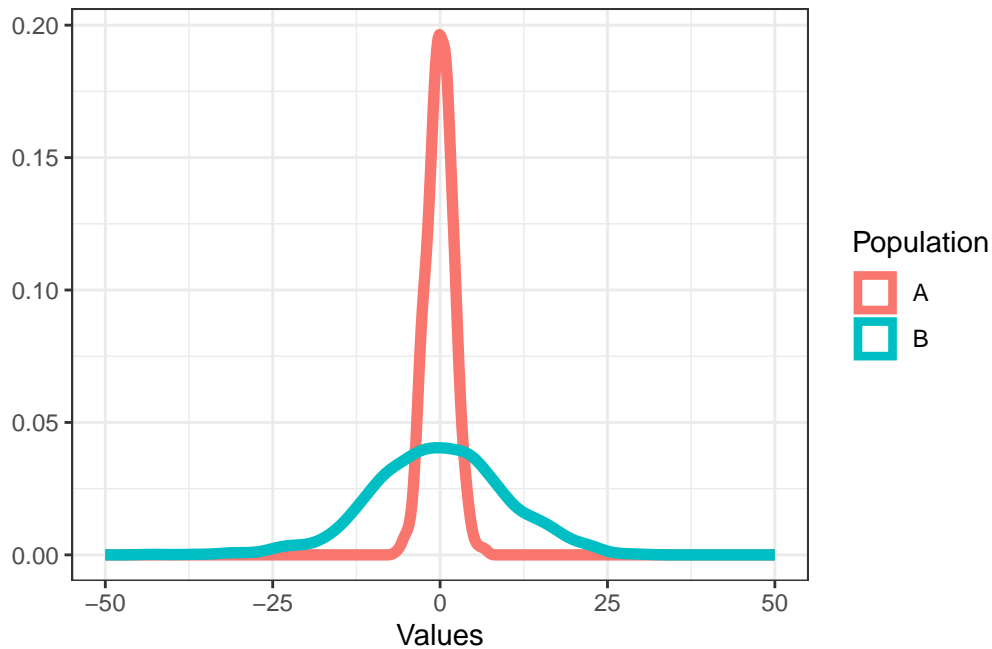
i Note

Smaller values of measures of dispersion indicate lower amount of dispersion.

4.2 Dispersion Example

Consider populations A and B shown below.

Warning: Using ``size`` aesthetic for lines was deprecated in ggplot2 3.4.0.
i Please use ``linewidth`` instead.



Where are the population of values centered? Which population of values has a higher amount of dispersion?

4.3 Measures of Dispersion

Commonly used measures of dispersion include:

- Range
- Variance
- Standard Deviation
- Coefficient of Variation
- Interquartile Range

4.4 Range

The range is the difference between the largest and smallest values in a set of observations.

$$R = x_L - x_S$$

where x_L and x_S are the largest and smallest values, respectively.

4.5 Range: Example

What is the range of the following values: 254, 281, 192, 260, 212, 179?

- The maximum value is 281, minimum value is 179. The range can be calculated as $281 - 179 = 102$.

4.6 Range: R

The `diff(range(vector))` function calculates the range of a set of values.

```
range(c(254, 281, 192, 260, 212, 179))
```

```
[1] 179 281
```

```
diff(range(c(254, 281, 192, 260, 212, 179)))
```

```
[1] 102
```

We can also use `max()` and `min()` to identify the maximum and minimum values. We can take the difference between the two values to calculate the range.

```
max(c(254, 281, 192, 260, 212, 179)) - min(c(254, 281, 192, 260, 212, 179))
```

```
[1] 102
```

4.7 Range: R Example

Consider the `flights` data set from the package `nycflights23`. The `distance` column contains the distance in miles between origin and destination airport for each flight. Calculate the range of the distances between origin and destination airports.

```
max(flights$distance)
```

```
[1] 4983
```

```
min(flights$distance)
```

```
[1] 80
```

```
max(flights$distance) - min(flights$distance)
```

```
[1] 4903
```

The range is 4,903 miles.

4.8 Exercise

Consider the `iris` data set that gives the measurements in centimeters of the variables sepal length and width and petal length and width, respectively, for 50 flowers from each of 3 species of iris.

4.8.1 Exercise

- What is the longest petal length in this data set?
- What is the shortest petal length in this data set?
- What is the range of the petal lengths?

4.8.2 Answer

```
max(iris$Petal.Length)
```

```
[1] 6.9
```

```
min(iris$Petal.Length)
```

```
[1] 1
```

```
max(iris$Petal.Length)-min(iris$Petal.Length)
```

```
[1] 5.9
```

4.9 Range: Advantages and Disadvantages

i Advantages

The main advantage in using the range is the simplicity of its computation. It also provides information on the span of the set of values.

i Disadvantages

The usefulness of the range is limited. The fact that it takes into account only two values causes it to be a poor measure of dispersion. It is often preferable to express the range as a number pair $[x_S, x_L]$, as is seen in some demographic tables in research papers.

4.10 Variance