Statistics and Probability

Topic 02 : Descriptive Statistics

Lecture 03 : Descriptive Statistics Methods for Categorical Data

Dr Kieran Murphy

Department of Department of Computing and Mathematics,
Waterford Institute of Technology.
(kmurphy@wit.ie)

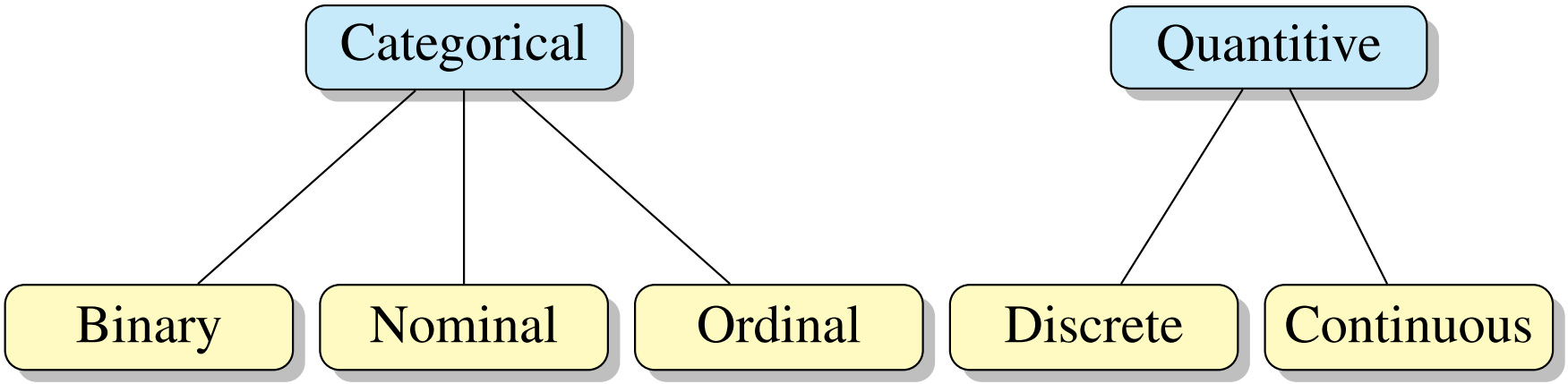
Spring Semester, 2017

Outline

- Numerical Methods — Frequency Counts, Mode
- Graphical Methods — Bar Chart, Pie Charts

Outline

Types of Data — Categorical Data

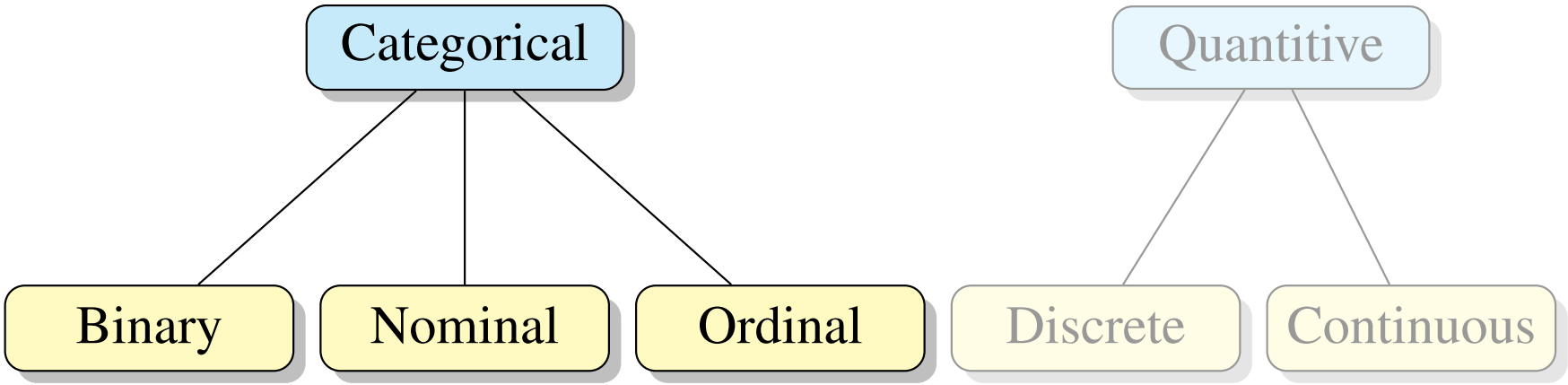


2 Categories	More Categories	Order matters	Numerical	+ Uninterrupted
TRUE/FALSE YES/NO, etc	Gender, Religion, Favourite Team, etc		**CAN** be represented by integers on the number line	Takes values from intervals onl the number line
			Number of students in this class	Duration of this class

Cannot order

Cannot add / subtract / multiple or divide

Types of Data — Categorical Data



2 Categories	More Categories	Order matters	Numerical	+ Uninterrupted
TRUE/FALSE YES/NO, etc	Gender, Religion, Favourite Team, etc		**CAN** be represented by integers on the number line	Takes values from intervals onl the number line
			Number of students in this class	Duration of this class

Cannot order

Cannot add / subtract / multiple or divide

$\Longrightarrow$ Limited to methods that only deal with counts/frequencies

Outline

Frequency/Count/Tally

Our first method, is trivial but widely applicable — we simply determine the frequency of each value in the data:

Method (Frequency/Count)

Count the number of occurrences of each distinct value in the data.

And our compulsory toy example:

Example 1

Summarise the following data

A, A, K, U, K, K, A, U, A, K, A, K, K, A, U, A, U, K, K, A, A, A, U, K, A

Label	Frequency
A	11
K	9
U	5

Frequency/Count/Tally

Our first method, is trivial but widely applicable — we simply determine the frequency of each value in the data:

Method (Frequency/Count)

Count the number of occurrences of each distinct value in the data.

And our compulsory toy example:

Example 1

Summarise the following data

A, A, K, U, K, K, A, U, A, K, A, K, K, A, U, A, U, K, K, A, A, A, U, K, A

Label	Frequency
A	11
K	9
U	5

Frequency/Count/Tally

Our first method, is trivial but widely applicable — we simply determine the frequency of each value in the data:

Method (Frequency/Count)

Count the number of occurrences of each distinct value in the data.

And our compulsory toy example:

Example 1

Summarise the following data

A, A, K, U, K, K, A, U, A, K, A, K, K, A, U, A, U, K, K, A, A, A, U, K, A

Label	Frequency
A	11
K	9
U	5

In R

R Commands

- table — build a contingency table of frequencies/counts.

frequency_example.R

```
x <- c("A", "A", "K", "U", "K", "K", "A", "U", "A", "K", "A",
       "K", "K", "A", "U", "A", "U", "K", "K", "A", "A", "A",
       "U", "K", "A")
table(x)
```

And output . . .

frequency_example.out

```
> x <- c("A", "A", "K", "U", "K", "K", "A", "U", "A", "K", "A",
+     "K", "K", "A", "U", "A", "U", "K", "K", "A", "A", "A",
+     "U", "K", "A")
> table(x)
x
 A  K  U
11  9  5
>
```

In R

R Commands

- table — build a contingency table of frequencies/counts.

frequency_example.R

```
1 x <- c("A", "A", "K", "U", "K", "K", "A", "U", "A", "K", "A",
2       "K", "K", "A", "U", "A", "U", "K", "K", "A", "A", "A",
3       "U", "K", "A")
4 table(x)
```

And output . . .

frequency_example.out

```
1 > x <- c("A", "A", "K", "U", "K", "K", "A", "U", "A", "K", "A",
2 +     "K", "K", "A", "U", "A", "U", "K", "K", "A", "A", "A",
3 +     "U", "K", "A")
4 > table(x)
5 x
6  A  K  U
7 11  9  5
8 >
```

In R

R Commands

- table — build a contingency table of frequencies/counts.

frequency_example.R

```
1 x <- c("A", "A", "K", "U", "K", "K", "A", "U", "A", "K", "A",
2       "K", "K", "A", "U", "A", "U", "K", "K", "A", "A", "A",
3       "U", "K", "A")
4 table(x)
```

And output . . .

frequency_example.out

```
1 > x <- c("A", "A", "K", "U", "K", "K", "A", "U", "A", "K", "A",
2 +     "K", "K", "A", "U", "A", "U", "K", "K", "A", "A", "A",
3 +     "U", "K", "A")
4 > table(x)
5 x
6  A  K  U
7 11  9  5
8 >
```

Relative Frequency

In order to aid the comparison of results we often work with relative frequency instead of frequency.

$$\text{relative frequency} = \frac{\text{frequency}}{\text{number of observations}}$$

Method (Relative Frequency)

Divide the frequency (number of occurrences) of each distinct value in the data by the number of observations in the data.

Example 2

There were 25 observations in the data in the previous example. Hence

Label	Frequency
A	11
K	9
U	5

$\Longrightarrow$

Label	Relative Frequency
A	0.44
K	0.36
U	0.20
	1.00

Relative Frequency

In order to aid the comparison of results we often work with relative frequency instead of frequency.

$$\text{relative frequency} = \frac{\text{frequency}}{\text{number of observations}}$$

Method (Relative Frequency)

Divide the frequency (number of occurrences) of each distinct value in the data by the number of observations in the data.

Example 2

There were 25 observations in the data in the previous example. Hence

Label	Frequency
A	11
K	9
U	5

$\Longrightarrow$

Label	Relative Frequency
A	0.44
K	0.36
U	0.20
	1.00

Relative Frequency

In order to aid the comparison of results we often work with relative frequency instead of frequency.

$$\text{relative frequency} = \frac{\text{frequency}}{\text{number of observations}}$$

Method (Relative Frequency)

Divide the frequency (number of occurrences) of each distinct value in the data by the number of observations in the data.

Example 2

There were 25 observations in the data in the previous example. Hence

Label	Frequency
A	11
K	9
U	5

$\Longrightarrow$

Label	Relative Frequency
A	0.44
K	0.36
U	0.20
	1.00

In R . . .

R Commands

- length — return the length of a vector.

relative _frequency_example.R

```
x <- c("A", "A", "K", "U", "K", "K", "A", "U", "A", "K", "A",
       "K", "K", "A", "U", "A", "U", "K", "K", "A", "A", "A",
       "U", "K", "A")
table(x)/length(x)
```

And output . . .

relative _frequency_example.out

```
> x <- c("A", "A", "K", "U", "K", "K", "A", "U", "A", "K", "A",
+    "K", "K", "A", "U", "A", "U", "K", "K", "A", "A", "A",
+    "U", "K", "A")
> table(x)/length(x)
x
   A    K    U
0.44 0.36 0.20
>
```

In R . . .

R Commands

- length — return the length of a vector.

relative _frequency_example.R

```
1 x <- c("A", "A", "K", "U", "K", "K", "A", "U", "A", "K", "A",
2       "K", "K", "A", "U", "A", "U", "K", "K", "A", "A", "A",
3       "U", "K", "A")
4 table(x)/length(x)
```

And output . . .

relative _frequency_example.out

```
1 > x <- c("A", "A", "K", "U", "K", "K", "A", "U", "A", "K", "A",
2 +     "K", "K", "A", "U", "A", "U", "K", "K", "A", "A", "A",
3 +     "U", "K", "A")
4 > table(x)/length(x)
5 x
6     A     K     U
7 0.44  0.36  0.20
8 >
```

In R . . .

R Commands

- length — return the length of a vector.

relative _frequency_example.R

```
1 x <- c("A", "A", "K", "U", "K", "K", "A", "U", "A", "K", "A",
2      "K", "K", "A", "U", "A", "U", "K", "K", "A", "A", "A",
3      "U", "K", "A")
4 table(x)/length(x)
```

And output . . .

relative _frequency_example.out

```
1 > x <- c("A", "A", "K", "U", "K", "K", "A", "U", "A", "K", "A",
2 +    "K", "K", "A", "U", "A", "U", "K", "K", "A", "A", "A",
3 +    "U", "K", "A")
4 > table(x)/length(x)
5 x
6    A    K    U
7 0.44 0.36 0.20
8 >
```

Comments

- The frequency, f, of a particular observation is the number of times the observation occurs in the data.
- The distribution of a variable is the pattern of frequencies of the observation.
- Frequency distributions can show either the actual number of observations falling in each range or the percentage of observations. In the latter instance:
  - The distribution is called a relative frequency distribution.
  - The sum of the relative frequencies is one.
- Computing the frequency distribution is often the first step in other methods.
- Frequency distribution tables can be used for both categorical and numeric* variables.

*Continuous variables should only be used with class intervals, which will covered in later in the module.

Mode

Definition 3 (Mode)

The mode is the value that appears most often in a data set.

Example 4

The mode of the data

A, A, K, U, K, K, A, U, A, K, A, K, K, A, U, A, U, K, K, A, A, A, U, K, A

has frequency distribution

Label	Frequency
A	11
K	9
U	5

Hence the mode is A (with frequency of 11).

In R . . . I

R does not have a function to calculate the mode directly because

- The mode is a relatively useless statistic.
- It can be calculated easily using R primitives.

mode_example.R

```
1 x <- c("A", "A", "K", "U", "K", "K", "A", "U", "A", "K", "A",
2     "K", "K", "A", "U", "A", "U", "K", "K", "A", "A", "A",
3     "U", "K", "A")
4 data <- table(x)
5 names(data)[data == max(data)]
```

And output . . .

mode_example.out

```
1 > x <- c("A", "A", "K", "U", "K", "K", "A", "U", "A", "K", "A",
2 +   "K", "K", "A", "U", "A", "U", "K", "K", "A", "A", "A",
3 +   "U", "K", "A")
4 > data <- table(x)
5 > names(data)[data == max(data)]
6 [1] "A"
7 >
```

In R . . . I

R does not have a function to calculate the mode directly because

- The mode is a relatively useless statistic.
- It can be calculated easily using R primitives.

mode_example.R

```
1 x <- c("A", "A", "K", "U", "K", "K", "A", "U", "A", "K", "A",
2     "K", "K", "A", "U", "A", "U", "K", "K", "A", "A", "A",
3     "U", "K", "A")
4 data <- table(x)
5 names(data)[data == max(data)]
```

And output . . .

mode_example.out

```
1 > x <- c("A", "A", "K", "U", "K", "K", "A", "U", "A", "K", "A",
2 +   "K", "K", "A", "U", "A", "U", "K", "K", "A", "A", "A",
3 +   "U", "K", "A")
4 > data <- table(x)
5 > names(data)[data == max(data)]
6 [1] "A"
7 >
```

⇐ Get name of column(s) with max frequency

In R . . . (Breakdown of calculating mode in R) II

mode_detail.out

```
1  > x <- c("A", "A", "K", "U", "K", "K", "A", "U", "A", "K", "A",
2  +    "K", "K", "A", "U", "A", "U", "K", "K", "A", "A", "A",
3  +    "U", "K", "A")
4  > data <- table(x)
5  > print(data)
6  x
7   A  K  U
8  11  9  5
9  > max(data)
10 [1] 11
11 > data == max(data)
12 x
13     A     K     U
14  TRUE FALSE FALSE
15 > data[data == max(data)]
16  A
17 11
18 > names(data)[data == max(data)]
19 [1] "A"
20 >
```

Line 4: Create table of frequencys.

Line 9: Determine max frequency.

Line 11: Locate columns with max frequency.

Line 15: Select column(s) with max frequency. (This is an example of logical indexing, which is frequently used in R to filter columns/rows in a dataset.)

Line 18: Get name of column(s) with max frequency.

In R . . . (Breakdown of calculating mode in R) II

mode_detail.out

```
1  > x <- c("A", "A", "K", "U", "K", "K", "A", "U", "A", "K", "A",
2  +    "K", "K", "A", "U", "A", "U", "K", "K", "A", "A", "A",
3  +    "U", "K", "A")
4  > data <- table(x)
5  > print(data)
6  x
7   A  K  U
8  11  9  5
9  > max(data)
10 [1] 11
11 > data == max(data)
12 x
13     A     K     U
14  TRUE FALSE FALSE
15 > data[data == max(data)]
16  A
17 11
18 > names(data)[data == max(data)]
19 [1] "A"
20 >
```

Line 4: Create table of frequencys.

Line 9: Determine max frequency.

Line 11: Locate columns with max frequency.

Line 15: Select column(s) with max frequency. (This is an example of logical indexing, which is frequently used in R to filter columns/rows in a dataset.)

Line 18: Get name of column(s) with max frequency.

In R . . . (Breakdown of calculating mode in R) II

mode_detail.out

```
1  > x <- c("A", "A", "K", "U", "K", "K", "A", "U", "A", "K", "A",
2  +    "K", "K", "A", "U", "A", "U", "K", "K", "A", "A", "A",
3  +    "U", "K", "A")
4  > data <- table(x)
5  > print(data)
6  x
7   A  K  U
8  11  9  5
9  > max(data)
10 [1] 11
11 > data == max(data)
12 x
13     A     K     U
14  TRUE FALSE FALSE
15 > data[data == max(data)]
16  A
17 11
18 > names(data)[data == max(data)]
19 [1] "A"
20 >
```

Line 4: Create table of frequencys.

Line 9: Determine max frequency.

Line 11: Locate columns with max frequency.

Line 15: Select column(s) with max frequency. (This is an example of logical indexing, which is frequently used in R to filter columns/rows in a dataset.)

Line 18: Get name of column(s) with max frequency.

In R ... (Breakdown of calculating mode in R) II

mode_detail.out

```
1  > x <- c("A", "A", "K", "U", "K", "K", "A", "U", "A", "K", "A",
2  +    "K", "K", "A", "U", "A", "U", "K", "K", "A", "A", "A",
3  +    "U", "K", "A")
4  > data <- table(x)
5  > print(data)
6  x
7   A  K  U
8  11  9  5
9  > max(data)
10 [1] 11
11 > data == max(data)
12 x
13     A     K     U
14  TRUE FALSE FALSE
15 > data[data == max(data)]
16  A
17 11
18 > names(data)[data == max(data)]
19 [1] "A"
20 >
```

Line 4: Create table of frequencys.

Line 9: Determine max frequency.

Line 11: Locate columns with max frequency.

Line 15: Select column(s) with max frequency. (This is an example of logical indexing, which is frequently used in R to filter columns/rows in a dataset.)

Line 18: Get name of column(s) with max frequency.

In R . . . (Breakdown of calculating mode in R) II

mode_detail.out

```
1  > x <- c("A", "A", "K", "U", "K", "K", "A", "U", "A", "K", "A",
2  +    "K", "K", "A", "U", "A", "U", "K", "K", "A", "A", "A",
3  +    "U", "K", "A")
4  > data <- table(x)
5  > print(data)
6  x
7   A  K  U
8  11  9  5
9  > max(data)
10 [1] 11
11 > data == max(data)
12 x
13    A     K     U
14  TRUE FALSE FALSE
15 > data[data == max(data)]
16  A
17 11
18 > names(data)[data == max(data)]
19 [1] "A"
20 >
```

Line 4: Create table of frequencys.

Line 9: Determine max frequency.

Line 11: Locate columns with max frequency.

Line 15: Select column(s) with max frequency. (This is an example of logical indexing, which is frequently used in R to filter columns/rows in a dataset.)

Line 18: Get name of column(s) with max frequency.

Comments

- If every value appears with the same frequency, then the mode is said not to exist.
- If two values appear at the same frequency, then the data is said to be bi-modal.
- More generally, if two or more values appear at the same frequency, then the data is said to be multimodal.
- We will discuss the mode in mode detail when dealing with quantitive data, when we will compare it to other measures of central tendency such as the median and the mean.

Outline

Barchart

Method (Bar chart)

Graphical representation of the frequency table for categorical data with separate vertical or horizontal bars for each value.
The length of the bars represents the frequency (or relative frequency).

Again, our toy example:

Example 5

Summarise the following data

A, A, K, U, K, K, A, U, A, K, A, K, K, A, U, A, U, K, K, A, A, A, U, K, A

Label	Frequency
A	11
K	9
U	5

Barchart

Method (Bar chart)

Graphical representation of the frequency table for categorical data with separate vertical or horizontal bars for each value.
The length of the bars represents the frequency (or relative frequency).

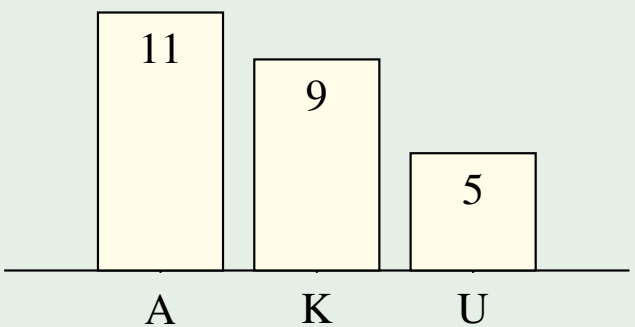
Again, our toy example:

Example 5

Summarise the following data

A, A, K, U, K, K, A, U, A, K, A, K, K, A, U, A, U, K, K, A, A, A, U, K, A

Label	Frequency
A	11
K	9
U	5



In R . . .

R Commands

- barplot — Creates a bar plot with vertical or horizontal bars.

To create a rudimentary bar chart is easy — we just build a table of frequencies and call barplot.

barchart_example_basic.R

```
1 x <- c("A", "A", "K", "U", "K", "K", "A", "U", "A", "K", "A",
2       "K", "K", "A", "U", "A", "U", "K", "K", "A", "A", "A",
3       "U", "K", "A")
4 barplot(table(x))
```

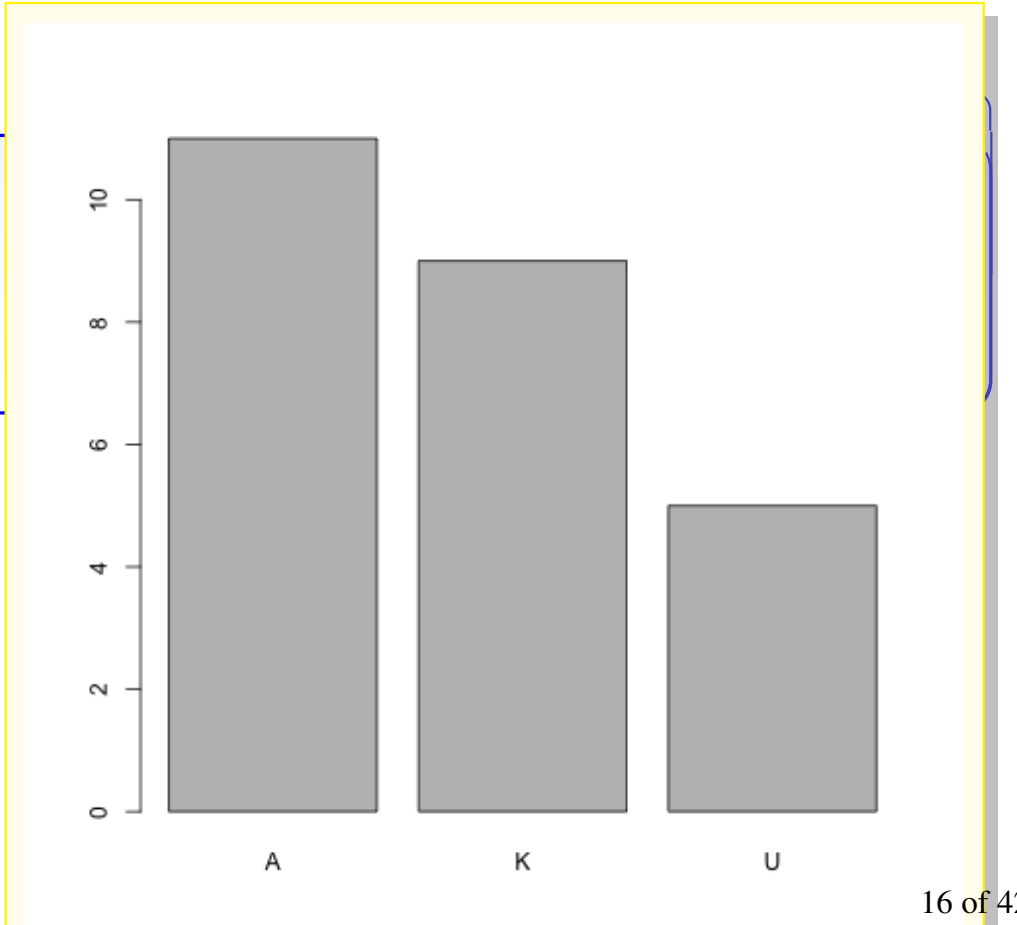
In R . . .

R Commands

- barplot — Creates a bar plot with vertical or horizontal bars.

To create a rudimentary bar chart is easy — we just build a table of frequencies and call barplot.

```
1 x <- c("A", "A", "K", "U", "K",
2       "K", "K", "A", "U", "A", "U"
3       "U", "K", "A")
4 barplot(table(x))
```



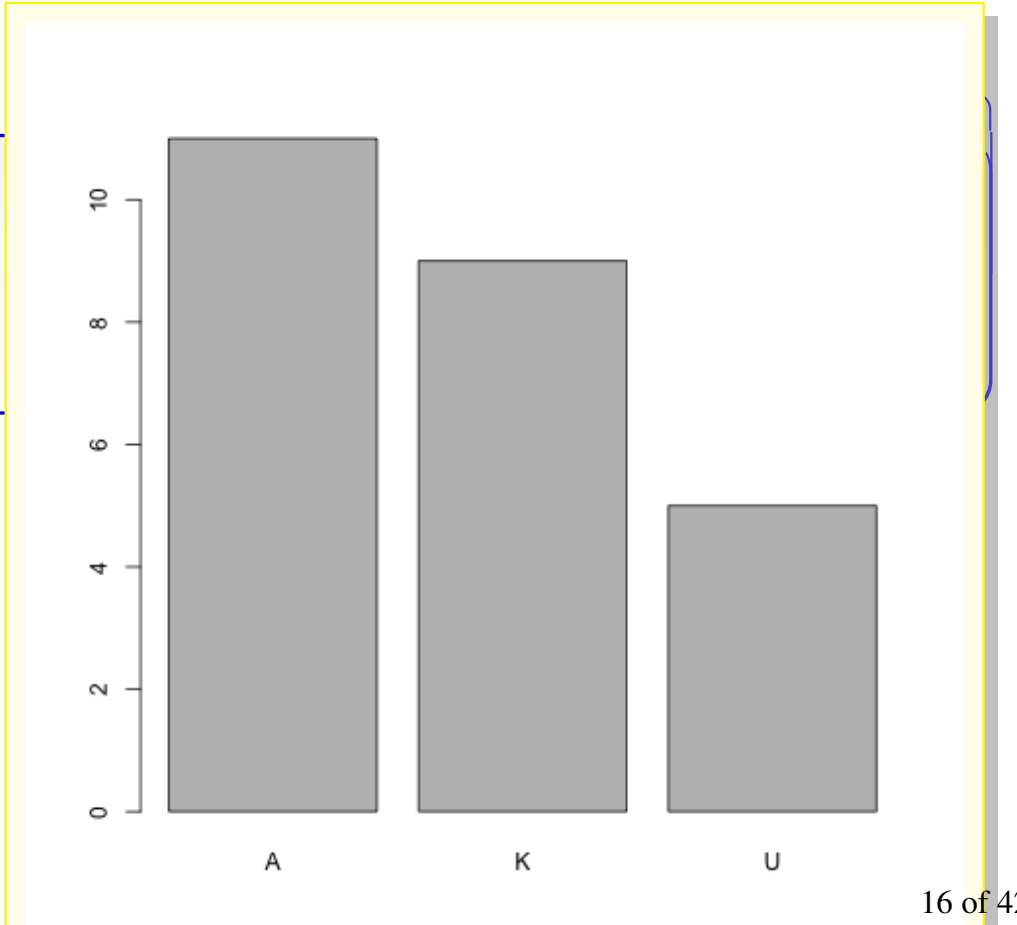
In R . . .

R Commands

- barplot — Creates a bar plot with vertical or horizontal bars.

To create a rudimentary bar chart is easy — we just build a table of frequencies and call barplot.

```
1 x <- c("A", "A", "K", "U", "K",
2       "K", "K", "A", "U", "A", "U"
3       "U", "K", "A")
4 barplot(table(x))
```



OK we have a bar chart . . . but it looks sad . . . and how do we save it?

Saving Graphical Output

Here we have (at least) three options:

- In RStudio, just click on `Export..Save as Image` and follow steps (use PNG or PDF format).
- Open a device for redirecting graphical output:

```
4 png(filename="barchart_example_save_only.png")
5 barplot(table(x))
6 dev.off()
```

- Save an existing image to a file:

```
4 barplot(table(x))
5
6 #dev.print(pdf, 'barchart_example_display_and_save.pdf')
```

Saving Graphical Output

Here we have (at least) three options:

- In RStudio, just click on `Export..Save as Image` and follow steps (use PNG or PDF format).
- Open a device for redirecting graphical output:

```
4 png(filename="barchart_example_save_only.png")
5 barplot(table(x))
6 dev.off()
```

- Save an existing image to a file:

```
4 barplot(table(x))
5
6 #dev.print(pdf, 'barchart_example_display_and_save.pdf')
```

Saving Graphical Output

Here we have (at least) three options:

- In RStudio, just click on `Export..Save as Image` and follow steps (use PNG or PDF format).
- Open a device for redirecting graphical output:

```
4 png(filename="barchart_example_save_only.png")
5 barplot(table(x))
6 dev.off()
```

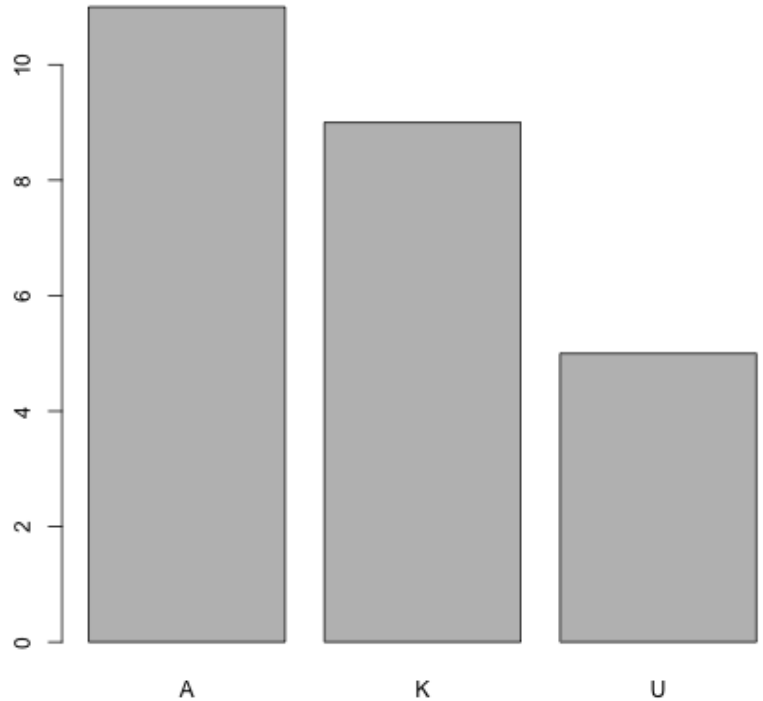
- Save an existing image to a file:

```
4 barplot(table(x))
5
6 #dev.print(pdf, 'barchart_example_display_and_save.pdf')
```

Pimping my Bar Chart — Starting Point

Issues

- Only one shade of grey.
- Missing x-, y- and main plot title.
- Font is too small for the size of the graph.
- Nice if we had labels on each column.



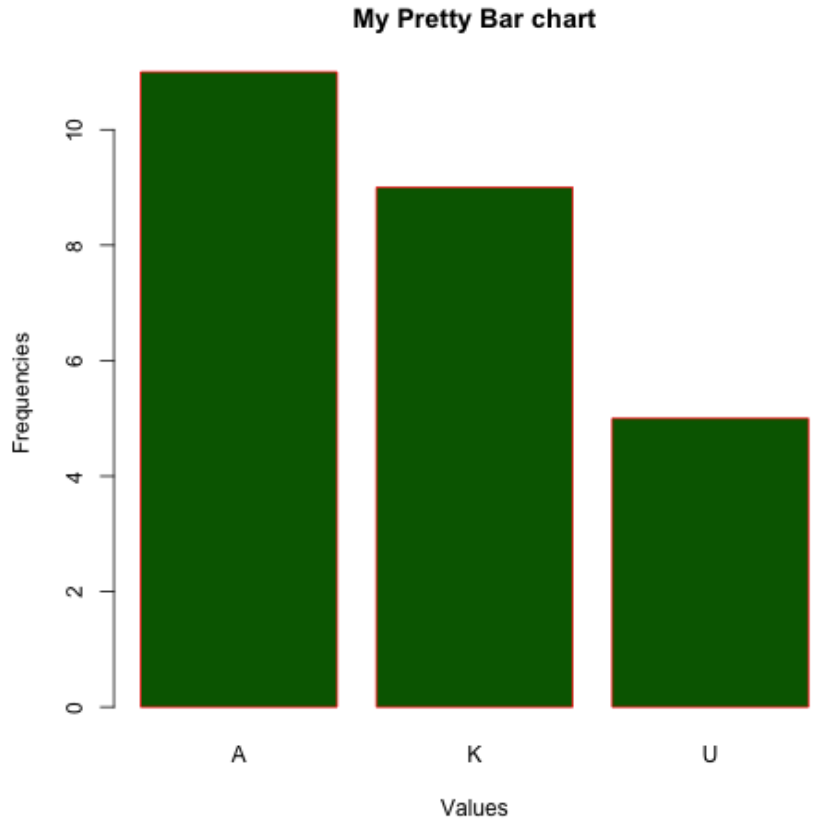
barchart_example_pimping.R

```
6 freq <- table(x)
7 barplot(freq)
```

Pimping my Bar Chart —Adding Titles and Some Colour

Use parameter

- main to set the main plot title.
- xlab and ylab to set the x- and y-axis labels respectively.
- border to set the border colour.
- col to set the colour, more on this later.



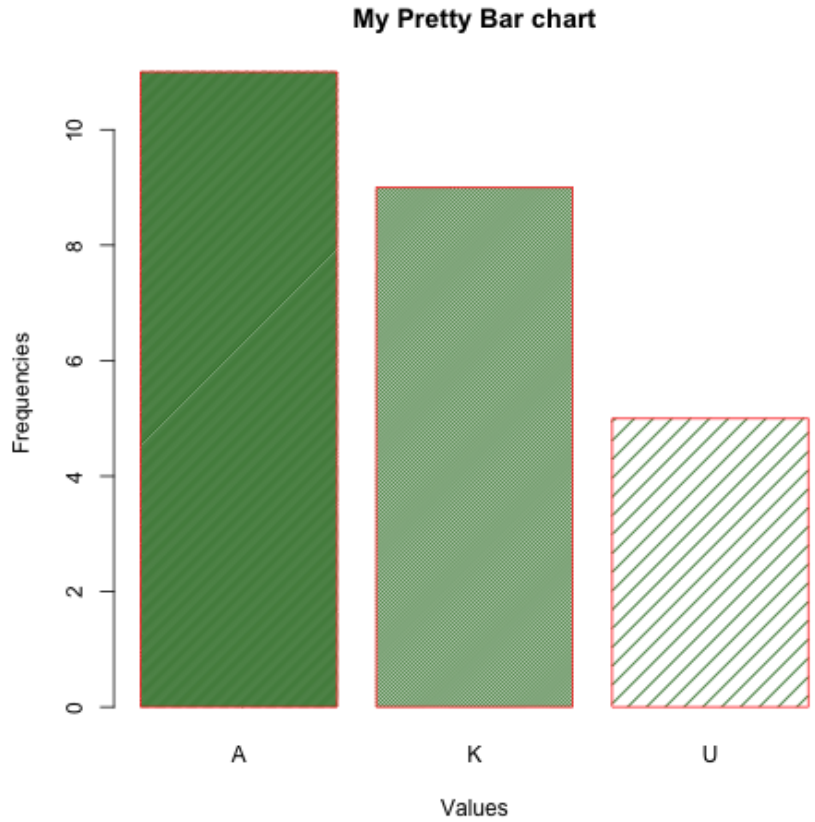
barchart_example_pimping.R

```
11 barplot(freq, main="My Pretty Bar chart",
12     xlab="Values", ylab="Frequencies",
13     col="darkgreen", border="red")
```

Pimping my Bar Chart — Shading

Use parameter

- density to set the shading for each bar. This is a value between 100 (opaque) and 0 (transparent).
  If density is set to a single value, then that value is used for all bars.
  If density is set to a vector, then its values are used cyclically over the bars.



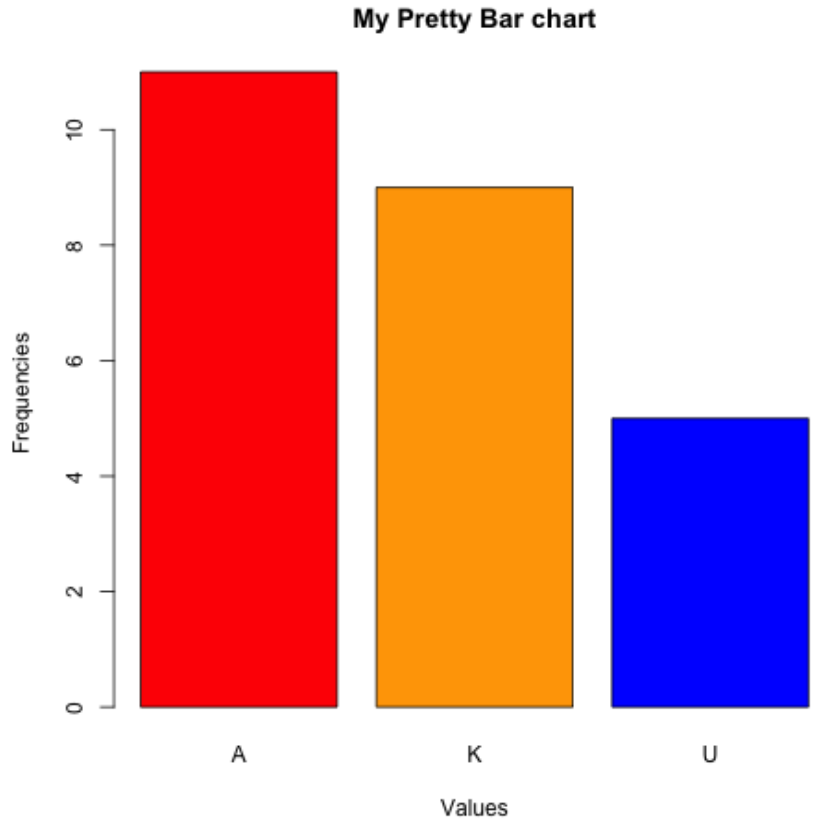
barchart_example_pimping.R

```
17 barplot(freq, main="My Pretty Bar chart",
18     xlab="Values", ylab="Frequencies",
19     col='darkgreen', border="red", density=c(90,50,10))
```

Pimping my Bar Chart — Colours

Use parameter

- col to set the colour for each bar.
  If col is set to a vector, then its values are used cyclically over the bars.



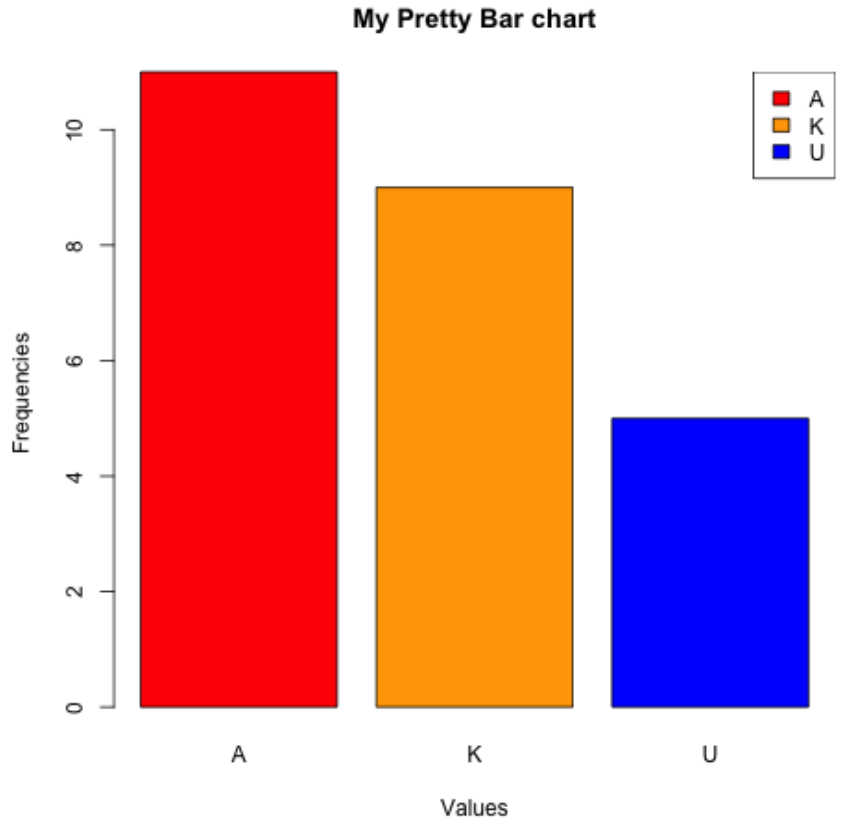
barchart_example_pimping.R

```
23 barplot(freq, main="My Pretty Bar chart",
24     xlab="Values", ylab="Frequencies",
25     col=c("red", "orange", "blue"))
```

Pimping my Bar Chart — Legend

Use command legend to add a legend to a graph. This command has parameters to

- specify position
- text used in labels (can be different from that used along x-axis
- Notice that since we wanted to use the list of colours for both the legend and the bar chart I first created a list of colours.



barchart_example_pimping.R

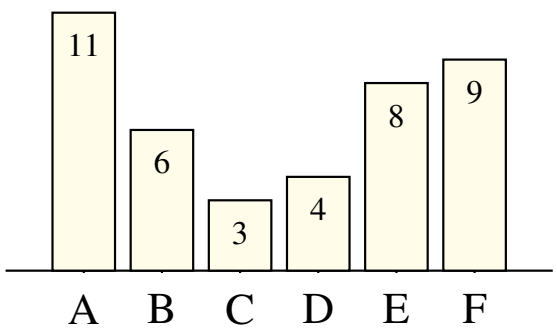
```
29 colors <- c("red", "orange", "blue", "yellow", "green")
30 barplot(freq, main="My Pretty Bar chart",
31     xlab="Values", ylab="Frequencies", col=colors)
32 legend("topright", names(freq), fill=colors)
```

Comments

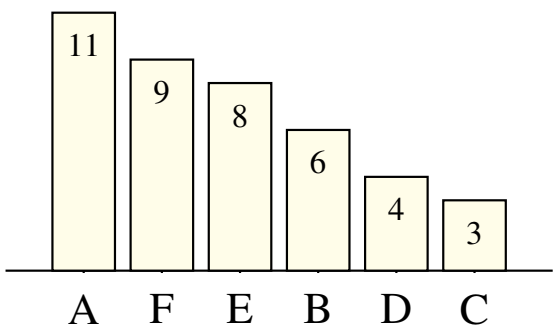
- It is of fundamental importance that a bar chart (graphical representation of categorical data) is not confused with a histogram (graphical representation of quantitive data).

  In a bar chart the bars are not touching.

- If we use relative frequencies in place of frequencies the bar chart will have the same shape but the scale on the vertical axis is changed.
- Also relative frequencies can often be represented as percentages.
- Since the order of the bars in a bar chart is arbitrary it does not make sense to speak of a distribution begin symmetrical or whether it is skewed.



is the same as



Pie Chart

Method (Pie Chart)

Divide a circle into regions based on the relative frequency of each value.

Again, our toy example:

Example 6

Construct a pie chart using our toy example

Label	Frequency	Rel. Frequency
A	11	0.44
K	9	0.36
U	5	0.20
		1.00

Pie Chart

Method (Pie Chart)

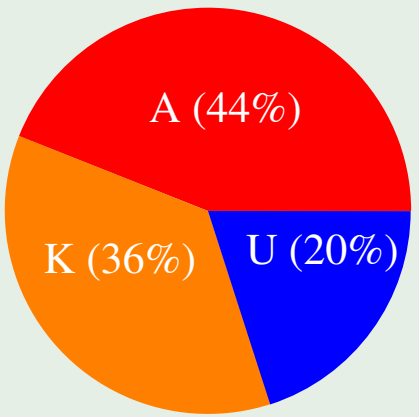
Divide a circle into regions based on the relative frequency of each value.

Again, our toy example:

Example 6

Construct a pie chart using our toy example

Label	Frequency	Rel. Frequency
A	11	0.44
K	9	0.36
U	5	0.20
		1.00



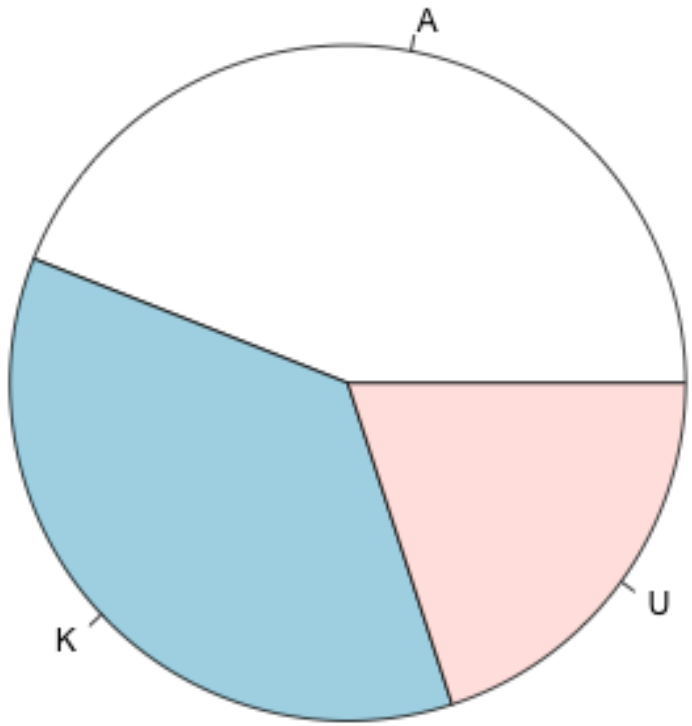
In R . . .

R Commands

- pie — Construct a pie chart.

piechart_example.R

```
5 freq <- table(x)
6 pie(freq)
```



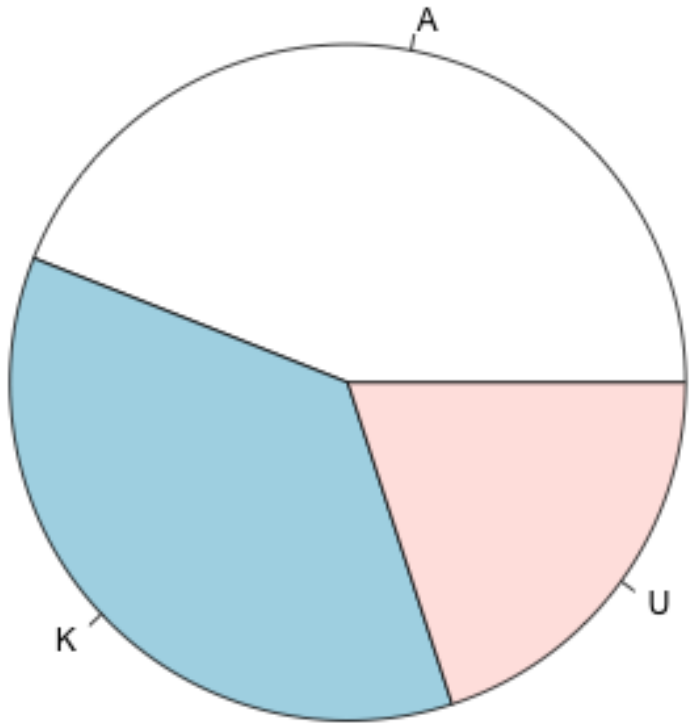
In R . . .

R Commands

- pie — Construct a pie chart.

piechart_example.R

```
5 freq <- table(x)
6 pie(freq)
```



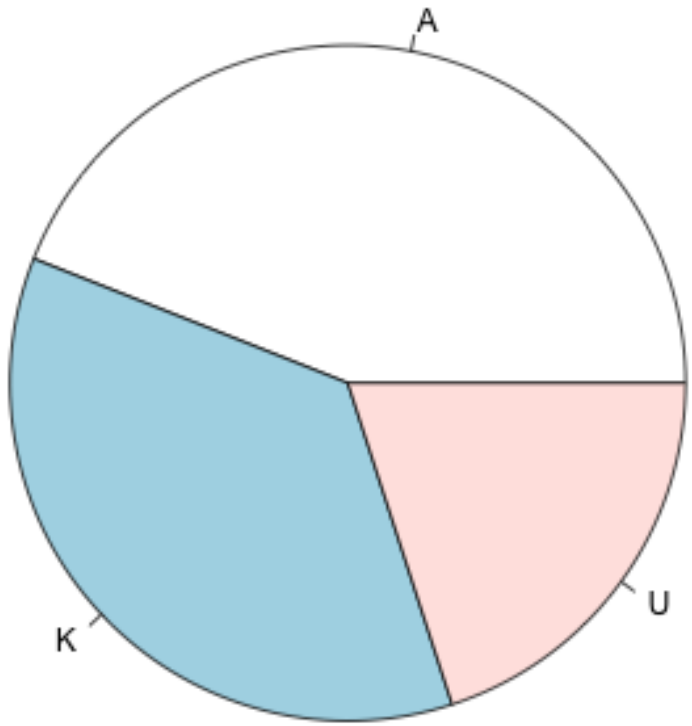
In R . . .

R Commands

- pie — Construct a pie chart.

piechart_example.R

```
5 freq <- table(x)
6 pie(freq)
```



This is our basic pie chart . . . we can pimp it as we did for the bar chart. In particular we SHOULD specify the frequencies OR the relative frequencies and the total number of observations.

In R . . .

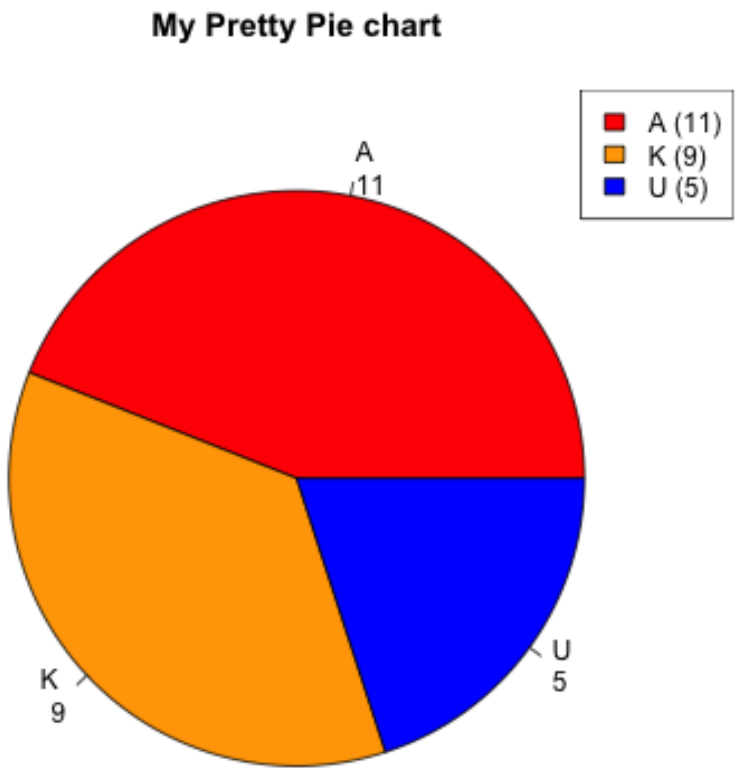
piechart_example.R

```
freq <- table(x)

colors <- c("red", "orange",
    "blue", "yellow", "green")
labels <- paste(names(freq)," ",
    "\n", freq, sep="")

pie(freq,
    main="My Pretty Pie chart",
    labels = labels,
    col=colors)

labels <- paste(names(freq),
    " (", freq, ")", sep="")
legend("topright", labels, fill=colors)
```



Comments

- Pie charts are often over used (especially by newspapers) but they have significant limitations:
  - Comparing two pie charts is problematic since people find it harder to compare angles to lengths.
  - If one uses relative frequencies (or percentages) then the number of observations needs to be clearly stated.
  - Should not be used if the number of categories is large.

To my mind, the best use of a pie chart is when you have one value that is overwhelmingly larger than the rest and you don't want the audience to focus on the actual values, but just bamboozle them with the overwhelming size of the leading segment. Of course, this seems to come close to embracing the old adage, "There are lies, damn lies and statistics."

— *www.juiceanalytics.com/writing/writing/the-problem-with-pie-charts*

See also

- www.edwardtufte.com/bboard/q-and-a-fetch-msg?msg_id=00018S

  (Edward Tufte is a statistician and author of 4 books on data visualisation.
- "Save the Pies for Dessert" article at www.perceptualedge.com/articles/08-21-07.pdf

Outline

The MASS Library — A Collection of Real Datasets I

OK, lets use our new statistical skills to study some real data. R comes with a collection of datasets that we can use:

Loading the library

First we load the library MASS, which contains the datasets and print a list of the available data sets.

```
1 library(MASS)          # MASS package is a collection of datasets
2 data()                 # show list of available datasets
```

```
> library(MASS)          # MASS package is a collection of datasets
> data()                 # show list of available datasets
Data sets in package datasets:

AirPassengers             Monthly Airline Passenger Numbers 1949-1960
BJsales                   Sales Data with Leading Indicator
BJsales.lead (BJsales)
                          Sales Data with Leading Indicator
BOD                       Biochemical Oxygen Demand
CO2                       Carbon Dioxide Uptake in Grass Plants
```

The MASS Library — A Collection of Real Datasets I

OK, lets use our new statistical skills to study some real data. R comes with a collection of datasets that we can use:

Loading the library

First we load the library MASS, which contains the datasets and print a list of the available data sets.

```
1 library(MASS)          # MASS package is a collection of datasets
2 data()                 # show list of available datasets
```

```
> library(MASS)          # MASS package is a collection of datasets
> data()                 # show list of available datasets
Data sets in package datasets:

AirPassengers             Monthly Airline Passenger Numbers 1949-1960
BJsales                   Sales Data with Leading Indicator
BJsales.lead (BJsales)
                          Sales Data with Leading Indicator
BOD                       Biochemical Oxygen Demand
CO2                       Carbon Dioxide Uptake in Grass Plants
```

The MASS Library — A Collection of Real Datasets I

OK, lets use our new statistical skills to study some real data. R comes with a collection of datasets that we can use:

Loading the library

First we load the library MASS, which contains the datasets and print a list of the available data sets.

```
1 library(MASS)          # MASS package is a collection of datasets
2 data()                 # show list of available datasets
```

```
> library(MASS)          # MASS package is a collection of datasets
> data()                 # show list of available datasets
Data sets in package datasets:

AirPassengers             Monthly Airline Passenger Numbers 1949-1960
BJsales                   Sales Data with Leading Indicator
BJsales.lead (BJsales)
                          Sales Data with Leading Indicator
BOD                       Biochemical Oxygen Demand
CO2                       Carbon Dioxide Uptake in Grass Plants
```

The MASS Library — A Collection of Real Datasets II

Selecting a Dataset

We will use the "survey" database which is a survey of 237 students. We first want to get some general information about the dataset.

```
3 help(survey)          # print a description of the dataset survey
```

```
> help(survey)          # print a description of the dataset survey
survey                    package:MASS
R Documentation

_S_t_u_d_e_n_t _S_u_r_v_e_y _D_a_t_a

_D_e_s_c_r_i_p_t_i_o_n:

     This data frame contains the responses of 237 Statistics I
     students at the University of Adelaide to a number of questions.

_U_s_a_g_e:

     survey
```

The MASS Library — A Collection of Real Datasets II

Selecting a Dataset

We will use the "survey" database which is a survey of 237 students. We first want to get some general information about the dataset.

```
3 help(survey)          # print a description of the dataset survey
```

```
> help(survey)          # print a description of the dataset survey
survey                    package:MASS
R Documentation

_S_t_u_d_e_n_t _S_u_r_v_e_y _D_a_t_a

_D_e_s_c_r_i_p_t_i_o_n:

     This data frame contains the responses of 237 Statistics I
     students at the University of Adelaide to a number of questions.

_U_s_a_g_e:

     survey
```

```
_F_o_r_m_a_t:

     The components of the data frame are:

     Sex The sex of the student. (Factor with levels "Male" and
         "Female".)

     Wr.Hnd span (distance from tip of thumb to tip of little finger
         of spread hand) of writing hand, in centimetres.

     NW.Hnd span of non-writing hand.

     W.Hnd writing hand of student. (Factor, with levels "Left" and
         "Right".)

     Fold Fold your arms! Which is on top (Factor, with levels "R
␣␣␣␣␣␣␣␣␣␣on␣L", "L␣on␣R", "Neither".)

     Pulse pulse rate of student (beats per minute).

     Clap Clap your hands!  Which hand is on top? (Factor, with
         levels "Right", "Left", "Neither".)
```

The MASS Library — A Collection of Real Datasets III

We load in the survey to access the data and get a list of the variables in the dataset

```
4 data(survey)          # load the dataset survey
5 str(survey)
```

```
> data(survey)          # load the dataset survey
> str(survey)
'data.frame':   237 obs. of  12 variables:
 $ Sex   : Factor w/ 2 levels "Female","Male": 1 2 2 2 2 1 2 1 2 2 ...
 $ Wr.Hnd: num  18.5 19.5 18 18.8 20 18 17.7 17 20 18.5 ...
 $ NW.Hnd: num  18 20.5 13.3 18.9 20 17.7 17.7 17.3 19.5 18.5 ...
 $ W.Hnd : Factor w/ 2 levels "Left","Right": 2 1 2 2 2 2 2 2 2 2 ...
 $ Fold  : Factor w/ 3 levels "L␣on␣R","Neither",..: 3 3 1 3 2 1 1 3 3
 $ Pulse : int  92 104 87 NA 35 64 83 74 72 90 ...
 $ Clap  : Factor w/ 3 levels "Left","Neither",..: 1 1 2 2 3 3 3 3 3 3
 $ Exer  : Factor w/ 3 levels "Freq","None",..: 3 2 2 2 3 3 1 1 3 3 ...
 $ Smoke : Factor w/ 4 levels "Heavy","Never",..: 2 4 3 2 2 2 2 2 2 2
 $ Height: num  173 178 NA 160 165 ...
 $ M.I   : Factor w/ 2 levels "Imperial","Metric": 2 1 NA 2 2 1 1 2 2
 $ Age   : num  18.2 17.6 16.9 20.3 23.7 ...
```

The MASS Library — A Collection of Real Datasets III

We load in the survey to access the data and get a list of the variables in the dataset

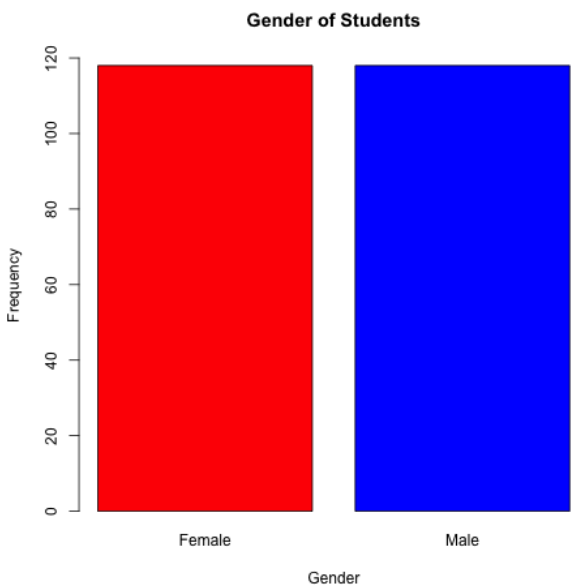
```
4 data(survey)          # load the dataset survey
5 str(survey)
```

```
> data(survey)          # load the dataset survey
> str(survey)
'data.frame':   237 obs. of  12 variables:
 $ Sex    : Factor w/ 2 levels "Female","Male": 1 2 2 2 2 1 2 1 2 2 ...
 $ Wr.Hnd: num  18.5 19.5 18 18.8 20 18 17.7 17 20 18.5 ...
 $ NW.Hnd: num  18 20.5 13.3 18.9 20 17.7 17.7 17.3 19.5 18.5 ...
 $ W.Hnd : Factor w/ 2 levels "Left","Right": 2 1 2 2 2 2 2 2 2 2 ...
 $ Fold   : Factor w/ 3 levels "L␣on␣R","Neither",..: 3 3 1 3 2 1 1 3 3
 $ Pulse : int  92 104 87 NA 35 64 83 74 72 90 ...
 $ Clap   : Factor w/ 3 levels "Left","Neither",..: 1 1 2 2 3 3 3 3 3 3
 $ Exer   : Factor w/ 3 levels "Freq","None",..: 3 2 2 2 3 3 1 1 3 3 ...
 $ Smoke : Factor w/ 4 levels "Heavy","Never",..: 2 4 3 2 2 2 2 2 2 2
 $ Height: num  173 178 NA 160 165 ...
 $ M.I    : Factor w/ 2 levels "Imperial","Metric": 2 1 NA 2 2 1 1 2 2
 $ Age    : num  18.2 17.6 16.9 20.3 23.7 ...
```

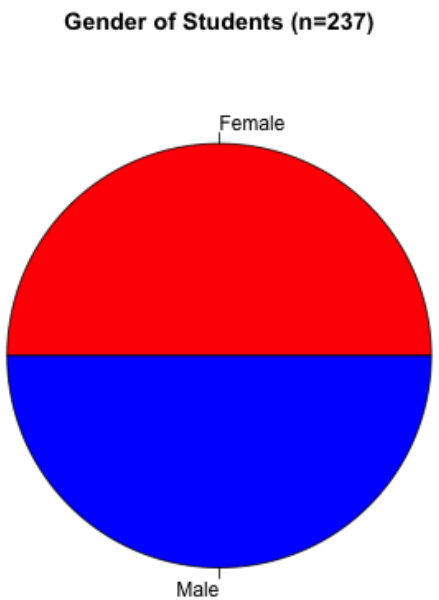
Analysing the Sex Variable


```
freq <- table(survey$Sex)
colors <- c("red", "blue", "orange", "yellow", "green")

barplot(freq, main="Gender of Students", ylim=c(0,120),
    xlab="Gender", ylab="Frequency", col=colors)
```



or




```
title <- sprintf("Gender of Students (n=%d)", length(survey$Sex))
pie(freq, main=title, labels=names(freq), col=colors)
```

Analysing the Sex Variable II

Well how good a job did ~~We~~ I do?

- In the bar chart it is hard to see the difference between the two groups — the two bars do appear to be of equal length.
- In the pie chart it is easier to see that, yes, the two regions are of equal size.
- But number of observations is odd so can't be split into two equal groups.

$\Longrightarrow$ So in summary, we have used up a lot of space with representations that are misleading (at worst) or incomplete (at best).

What should you have done instead?

I would have just generated a sentence of the form:

> The NUM observations consists of NUM (PERCENTAGE) Females, and NUM (PERCENTAGE) Males.

Analysing the Sex Variable II

Well how good a job did ~~We~~ I do?

- In the bar chart it is hard to see the difference between the two groups — the two bars do appear to be of equal length.
- In the pie chart it is easier to see that, yes, the two regions are of equal size.
- But number of observations is odd so can't be split into two equal groups.

$\Longrightarrow$ So in summary, we have used up a lot of space with representations that are misleading (at worst) or incomplete (at best).

What should you have done instead?

I would have just generated a sentence of the form:

> The NUM observations consists of NUM (PERCENTAGE) Females, and NUM (PERCENTAGE) Males.

Analysing the Sex Variable II

Well how good a job did ~~We~~ I do?

- In the bar chart it is hard to see the difference between the two groups — the two bars do appear to be of equal length.
- In the pie chart it is easier to see that, yes, the two regions are of equal size.
- But number of observations is odd so can't be split into two equal groups.

$\Longrightarrow$ So in summary, we have used up a lot of space with representations that are misleading (at worst) or incomplete (at best).

What should you have done instead?

I would have just generated a sentence of the form:

> The NUM observations consists of NUM (PERCENTAGE) Females, and NUM (PERCENTAGE) Males.

Analysing the Sex Variable III

Lets look at the actual data (always a good step)

```
22 survey$Sex
```

```
  [1] Female Male   Male   Male   Male   Female Male   Female Male
Male
 [11] Female Male   Female Female Male   Female Female Male   Male
Male
 [21] Male   Male   Male   Male   Female Male   Male   Male   Male
Male
 [31] Female Male   Female Male   Male   Male   Female Male   Male
Male
 [41] Female Female Male   Female Female Male   Male   Male
Female Female
 [51] Male   Male   Male   Male   Male   Male   Female Male   Male
Male
 [61] Male   Female Female Female Female Male   Female Female Male
Male
 [71] Female Male   Female Female Female Female Female Female Female Male
 [81] Male   Male   Female Female Male   Female Female Female Male
Female
 [91] Male   Female Female Female Male   Female Male   Female Male
Female
```

Analysing the Sex Variable III

Lets look at the actual data (always a good step)

```
22 survey$Sex
```

```
  [1] Female Male   Male   Male   Male   Female Male   Female Male
Male
 [11] Female Male   Female Female Male   Female Female Male   Male
Male
 [21] Male   Male   Male   Male   Female Male   Male   Male   Male
Male
 [31] Female Male   Female Male   Male   Male   Female Male   Male
Male
 [41] Female Female Male   Female Female Male   Male   Male
Female Female
 [51] Male   Male   Male   Male   Male   Male   Female Male   Male
Male
 [61] Male   Female Female Female Female Male   Female Female Male
Male
 [71] Female Male   Female Female Female Female Female Female Female Male
 [81] Male   Male   Female Female Male   Female Female Female Male
Female
 [91] Male   Female Female Female Male   Female Male   Female Male
Female
```

Missing Values I

- In real data, we frequently have incomplete datasets, and have to deal with observations for which some variables are not specified. These are called missing values, and in R are indicated by NA (for Not Available).
- We could:
  - Delete any observation that contains a missing value
    - Easy to do . . .
    - Can result in deleting too many observations
  - Instead, best practice, is to keep all of the observations and deal with missing values individually when the need arises.

Missing Values I

- In real data, we frequently have incomplete datasets, and have to deal with observations for which some variables are not specified. These are called missing values, and in R are indicated by NA (for Not Available).
- We could:
  - Delete any observation that contains a missing value
    - ✔ Easy to do . . .

```
24 newData <- na.omit(survey)
```

    - ✘ Can result in deleting too many observations.

```
26 c(nrow(survey), nrow(newData))
```

```
> c(nrow(survey), nrow(newData))
[1] 237 168
```

  - Instead, best practice, is to keep all of the observations and deal with missing values individually when the need arises.

Missing Values I

- In real data, we frequently have incomplete datasets, and have to deal with observations for which some variables are not specified. These are called missing values, and in R are indicated by NA (for Not Available).
- We could:
  - Delete any observation that contains a missing value
    - ✔ Easy to do . . .

```
24 newData <- na.omit(survey)
```

    - ✘ Can result in deleting too many observations.

```
26 c(nrow(survey), nrow(newData))
```

```
> c(nrow(survey), nrow(newData))
[1] 237 168
```

  - Instead, best practice, is to keep all of the observations and deal with missing values individually when the need arises.

Missing Values I

- In real data, we frequently have incomplete datasets, and have to deal with observations for which some variables are not specified. These are called missing values, and in R are indicated by NA (for Not Available).
- We could:
  - Delete any observation that contains a missing value
    - ✔ Easy to do . . .

```
24 newData <- na.omit(survey)
```

    - ✘ Can result in deleting too many observations.

```
26 c(nrow(survey), nrow(newData))
```

```
> c(nrow(survey), nrow(newData))
[1] 237 168
```

  - Instead, best practice, is to keep all of the observations and deal with missing values individually when the need arises.

Missing Values I

- In real data, we frequently have incomplete datasets, and have to deal with observations for which some variables are not specified. These are called missing values, and in R are indicated by NA (for Not Available).
- We could:
  - Delete any observation that contains a missing value
    - ✔ Easy to do . . .

```
24 newData <- na.omit(survey)
```

    - ✘ Can result in deleting too many observations.

```
26 c(nrow(survey), nrow(newData))
```

```
> c(nrow(survey), nrow(newData))
[1] 237 168
```

  - Instead, best practice, is to keep all of the observations and deal with missing values individually when the need arises.

Missing Values I

- In real data, we frequently have incomplete datasets, and have to deal with observations for which some variables are not specified. These are called missing values, and in R are indicated by NA (for Not Available).
- We could:
  - Delete any observation that contains a missing value
    - ✔ Easy to do . . .

```
24 newData <- na.omit(survey)
```

    - ✘ Can result in deleting too many observations.

```
26 c(nrow(survey), nrow(newData))
```

```
> c(nrow(survey), nrow(newData))
[1] 237 168
```

  - Instead, best practice, is to keep all of the observations and deal with missing values individually when the need arises.

Missing Values II

- Rather than using table (which drops observations with missing values), we use summary (which includes the observations with missing values).

```
28 table(survey$Sex)
29 summary(survey$Sex)
```

```
> table(survey$Sex)

Female   Male
   118    118
> summary(survey$Sex)
Female   Male   NA's
   118    118      1
```

- We could count the number of missing values

```
31 sum(is.na(survey$Sex))
```

Missing Values II

- Rather than using table (which drops observations with missing values), we use summary (which includes the observations with missing values).

```
28 table(survey$Sex)
29 summary(survey$Sex)
```

```
> table(survey$Sex)

Female   Male
   118    118
> summary(survey$Sex)
Female   Male   NA's
   118    118      1
```

- We could count the number of missing values

```
31 sum(is.na(survey$Sex))
```

```
> sum(is.na(survey$Sex))
[1] 1
```

Analysing the Sex Variable III

Returning to generating the desired sentence we could do something like

```
33 freq <- summary(survey$Sex)
34 sprintf("The␣%d␣observations␣consists␣of␣%d␣(%.1f%%)␣Females␣\
35 ␣␣␣␣and␣%d␣(%.1f%%)␣Males.␣(And␣%d␣missing␣values)",
36       nrow(survey),
37       freq["Female"], 100*freq["Female"]/nrow(survey),
38       freq["Male"], 100*freq["Male"]/nrow(survey),
39       freq["NA's"])
```

which generates the sentence

The 237 observations consists of 118 (49.8%) Females and 118 (49.8%) Males. (And 1 missing values)

Relationship Between Two Categorical Variables I

We can use table and barplot commands to examine whether two categorical variables are related.
For example, lets see if there is a relationship between gender (represented by variable Sex) and preferred writing hand (represented by W.Hnd).

Filter the Dataset

First step is to create a reduced dataset containing only the two variables that we are interested in and drop any rows that contain missing values.

```
44 tmp <- na.omit(survey[c("Sex", 'W.Hnd')])
45 colors <- c("red", "blue", "orange", "yellow", "green")
```

Construct the Two-Way Frequency Table

To create a two-way frequency table we call table with the two variables.
The order is important.

Relationship Between Two Categorical Variables I

We can use table and barplot commands to examine whether two categorical variables are related.
For example, lets see if there is a relationship between gender (represented by variable Sex) and preferred writing hand (represented by W.Hnd).

Filter the Dataset

First step is to create a reduced dataset containing only the two variables that we are interested in and drop any rows that contain missing values.

```
44 tmp <- na.omit(survey[c("Sex", 'W.Hnd')])
45 colors <- c("red", "blue", "orange", "yellow", "green")
```

Construct the Two-Way Frequency Table

To create a two-way frequency table we call table with the two variables.
The order is important.

Relationship Between Two Categorical Variables I

We can use table and barplot commands to examine whether two categorical variables are related.
For example, lets see if there is a relationship between gender (represented by variable Sex) and preferred writing hand (represented by W.Hnd).

Filter the Dataset

First step is to create a reduced dataset containing only the two variables that we are interested in and drop any rows that contain missing values.

```
44 tmp <- na.omit(survey[c("Sex", 'W.Hnd')])
45 colors <- c("red", "blue", "orange", "yellow", "green")
```

Construct the Two-Way Frequency Table

To create a two-way frequency table we call table with the two variables.
The order is important.

Relationship Between Two Categorical Variables II

```
47 freq <- table(tmp$Sex, tmp$W.Hnd)
```

and

```
50 freq <- table(tmp$W.Hnd, tmp$Sex)
```

Construct the Barplot

When building a bar chart with two categorical variables we can specify the bars to atop each other (the default) or grouped side-by-side (using parameter beside=TRUE)

```
60 freq <- table(tmp$Sex, tmp$W.Hnd)
61 barplot(freq, main="Gender by Writing Hand", col=colors[1:2],
```

Relationship Between Two Categorical Variables II

```
47 freq <- table(tmp$Sex, tmp$W.Hnd)
```

```
        Left Right
Female     7   110
Male      10   108
```

and

```
50 freq <- table(tmp$W.Hnd, tmp$Sex)
```

```
       Female Male
Left        7   10
Right     110  108
```

Construct the Barplot

When building a bar chart with two categorical variables we can specify the bars to atop each other (the default) or grouped side-by-side (using parameter beside=TRUE)

```
60 freq <- table(tmp$Sex, tmp$W.Hnd)
61 barplot(freq, main="Gender by Writing Hand", col=colors[1:2],
```

Relationship Between Two Categorical Variables II

```
47 freq <- table(tmp$Sex, tmp$W.Hnd)
```

```
        Left Right
Female     7   110
Male      10   108
```

and

```
50 freq <- table(tmp$W.Hnd, tmp$Sex)
```

```
       Female Male
Left        7   10
Right     110  108
```

Construct the Barplot

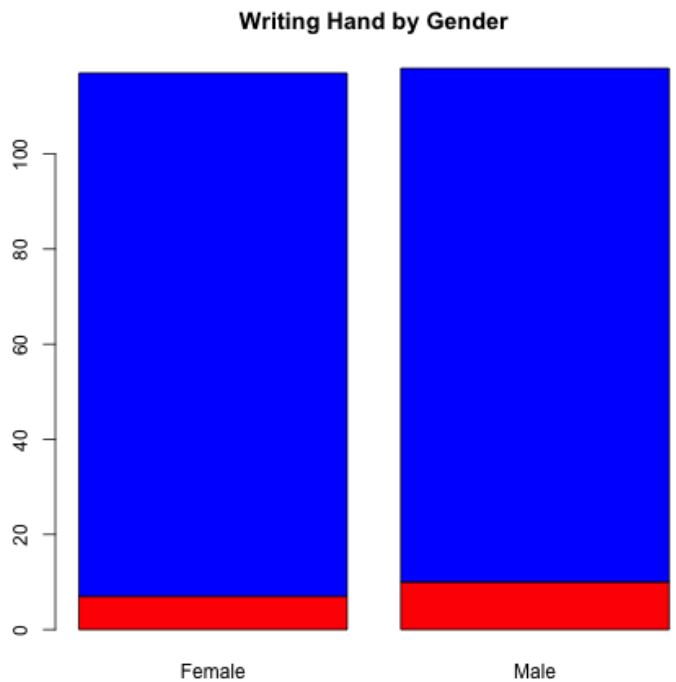
When building a bar chart with two categorical variables we can specify the bars to atop each other (the default) or grouped side-by-side (using parameter beside=TRUE)

```
60 freq <- table(tmp$Sex, tmp$W.Hnd)
61 barplot(freq, main="Gender by Writing Hand", col=colors[1:2],
```

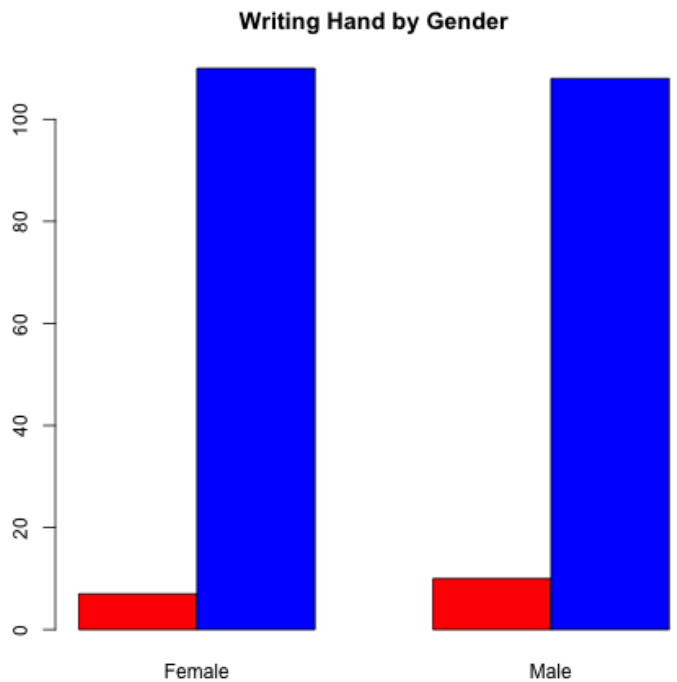
Relationship Between Two Categorical Variables IV

```
50 freq <- table(tmp$W.Hnd, tmp$Sex)
```

beside=FALSE



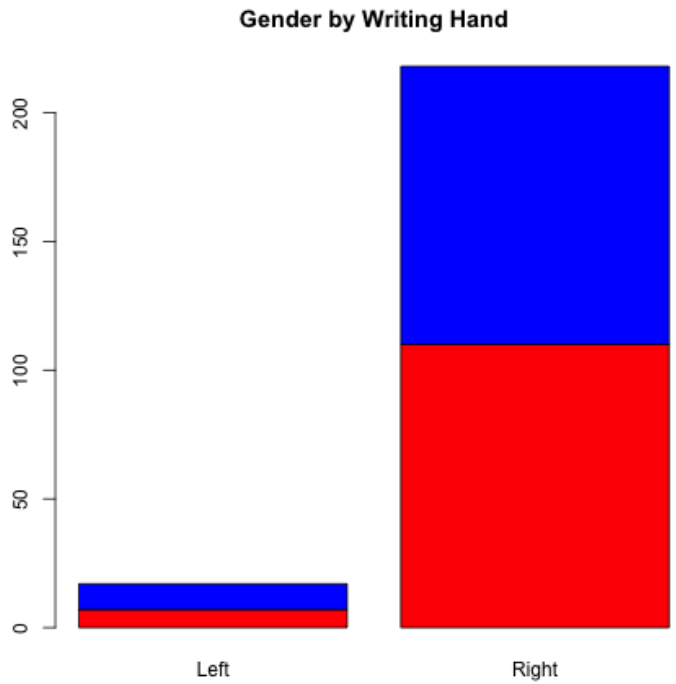
beside=TRUE



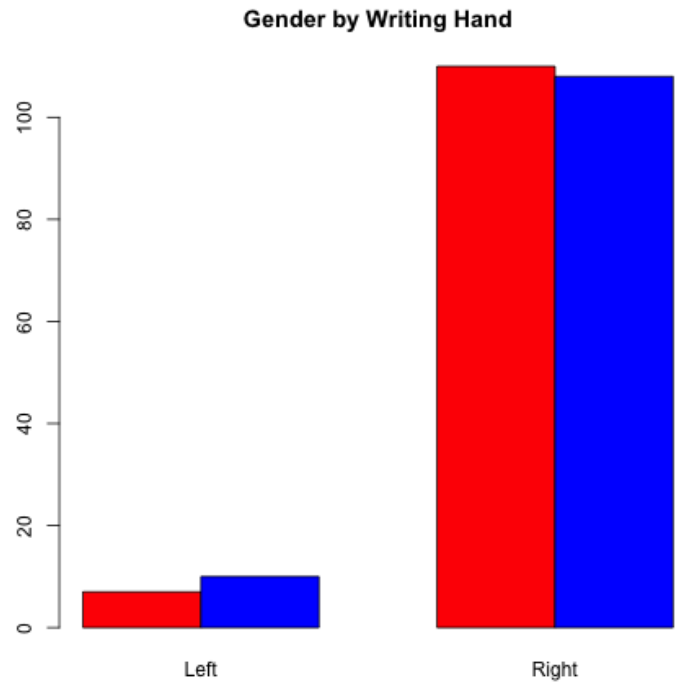
Relationship Between Two Categorical Variables IV

```
47 freq <- table(tmp$Sex, tmp$W.Hnd)
```

beside=FALSE



beside=TRUE



The picture here is a little clearer — especially for the beside=TRUE version.