1 Exploratory data analysis - Numerical summaries

1.1 Middle of a dataset (sample mean and median)

Both methods have pros and cons. The sample mean is the natural analogue for a dataset of what the expectation is for a probability distribution. However, it is very sensitive to outliers, by which we mean observations in the dataset that deviate a lot from the bulk of the data.

See the section *[Sample mean]* for how to calculate it in `R`.

1.1.1 Sample median

We can calulate the sample median using the `median()` function like this:

```
# define values
vals <- c(3, -2, -5, 2, 5, 2, 5, -1, -3, 4, 2)

#calculate sample median
median(vals)
```

```
## [1] 2
```

1.2 The amount of variability of a dataset

See the section *[Sample variance]* for how to calculate it in `R`.

1.2.1 Sample standard deviation

We can calculate the sample standar deviation using the `sd()` function like this:

```
#create dataset
data <- c(1, 3, 4, 6, 11, 14, 17, 20, 22, 23)

#find standard deviation
sd(data)
```

```
## [1] 8.279157
```

Note that you must use na.rm = TRUE to calculate the standard deviation if there are missing values in the dataset:

```
#create dataset with missing values
data <- c(1, 3, 4, 6, NA, 14, NA, 20, 22, 23)

#attempt to find standard deviation
sd(data)
```

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

```
#find standard deviation and specify to ignore missing values
sd(data, na.rm = TRUE)
```

```
## [1] 9.179753
```

A more robust measure of variability is the median of absolute deviations or MAD. The median absolute deviation measures the spread of observations in a dataset.

It's a particularly useful metric because it's less affected by outliers than other measures of dispersion like standard deviation and variance.

The formula to calculate median absolute deviation, often abbreviated MAD, is as follows:

$$MAD = median(|x_i - x_m|)$$

where:

- x_i: The ith value in the dataset
- x_m: The median value in the dataset

The following examples shows how to calculate the median absolute deviation in R by using the built-in `mad()` function.

The following code shows how to calculate the median absolute deviation for a single vector in R:

```
#define data
data <- c(1, 4, 4, 7, 12, 13, 16, 19, 22, 24)

#calculate MAD
mad(data)
```

```
## [1] 11.1195
```

1.3 Empirical quantiles, quartiles, and the IQR

Instead of identifying only the center of the dataset, *Tukey* suggested to give a five-number summary of the dataset: the minimum, the maximum, the sample median, and the 25th and 75th empirical percentiles.

The 25th empirical percentile $q_n(0.25)$ is called the *lower quartile* and the 75th empirical percentile $q_n(0.75)$ is called the *upper quartile*.

Together with the median, the lower and upper quartiles divide the dataset in four more or less equal parts consisting of about one quarter of the number of elements.

1.3.1 `quantile(x, probs = seq(0, 1, 0.25), na.rm = FALSE)`

In statistics, quantiles are values that divide a ranked dataset into equal groups.

The `quantile()` function in `R` can be used to calculate sample quantiles of a dataset where:

- `x`: Name of vector
- `probs`: Numeric vector of probabilities
- `na.rm`: Whether to remove NA values

```
#define vector of data
data <- c(1, 3, 3, 4, 5, 7, 8, 9, 12, 13, 13, 15, 18, 20, 22, 23, 24, 28)

#calculate quartiles
quantile(data, probs = seq(0, 1, 1 / 4))
```

```
##   0%  25%  50%  75% 100%
##  1.0  5.5 12.5 19.5 28.0
```

```
#calculate quintiles
quantile(data, probs = seq(0, 1, 1 / 5))
```

```
##   0%  20%  40%  60%  80% 100%
##  1.0  4.4  8.8 13.4 21.2 28.0
```

```
#calculate deciles
quantile(data, probs = seq(0, 1, 1 / 10))
```

```
##   0%  10%  20%  30%  40%  50%  60%  70%  80%  90% 100%
##  1.0  3.0  4.4  7.1  8.8 12.5 13.4 17.7 21.2 23.3 28.0
```

```
#calculate random quantiles of interest
quantile(data, probs = c(.2, .5, .9))
```

```
##  20%  50%  90%
##  4.4 12.5 23.3
```

The distance between the upper and lower quartiles is called the interquartile range, or IQR. You can also get the five number summary in R by using the summary() function:

```
summary(data)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    1.00    5.50   12.50   12.67   19.50   28.00
```

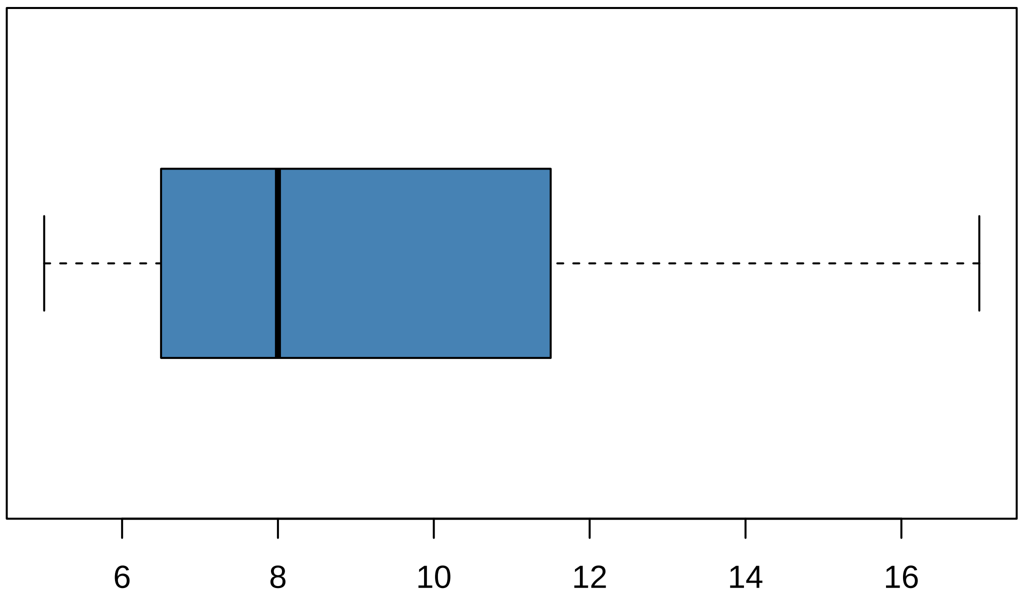
Which as you can see is the same as the quartiles calculated above. This five number summary can also be visualized as a boxplot.

1.3.2 Boxplots

To create a boxplot in R you can simply use the boxplot() function

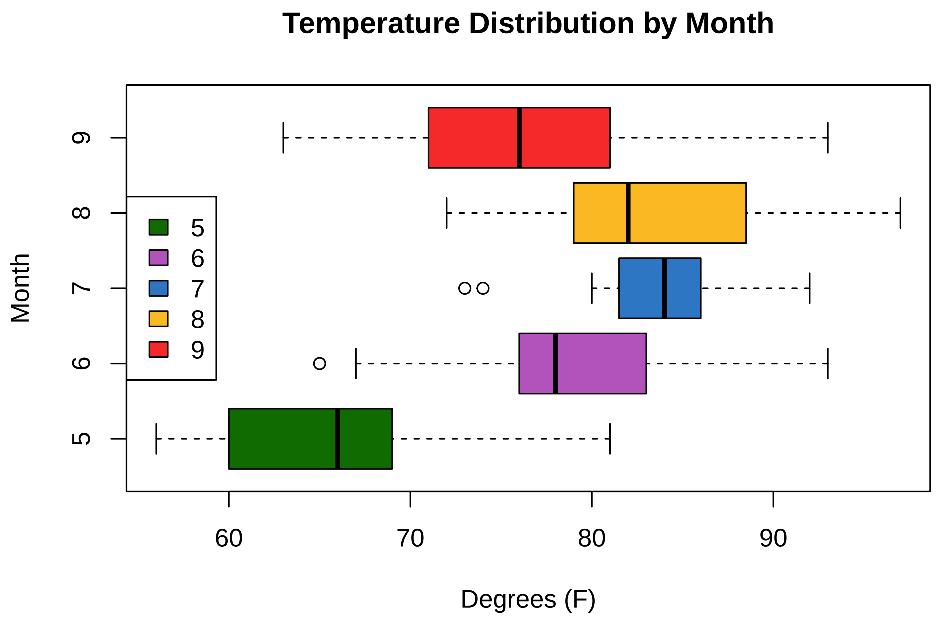
```
#create data
df <- data.frame(points = c(7, 8, 9, 12, 12, 5, 6, 6, 8, 11, 6, 8, 9, 13, 17),
                 team = rep(c('A', 'B', 'C'), each = 5))

#create horizontal boxplot for points
#Note: without the 'horizontal = TRUE' the boxplot would simply be vertical
boxplot(df$points, horizontal = TRUE, col = 'steelblue')
```



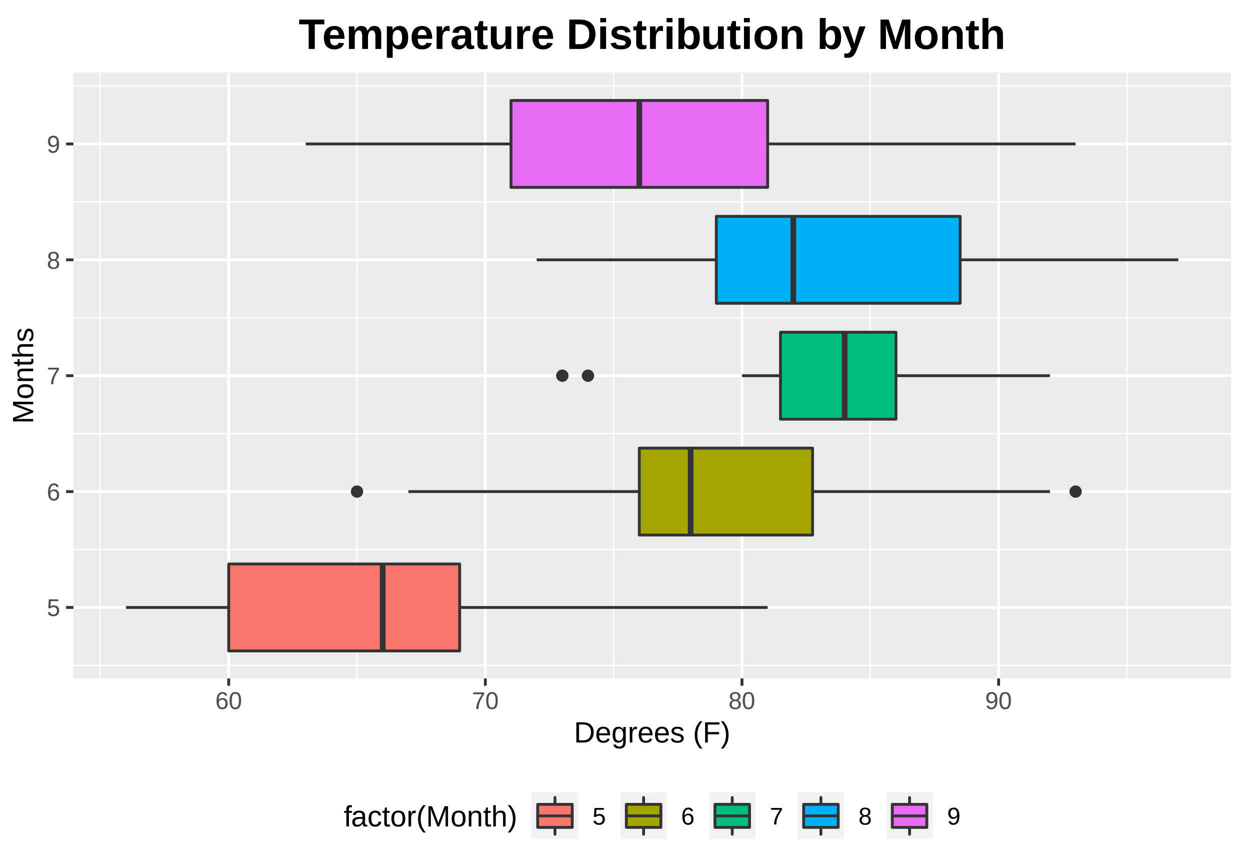
You can also put multiple boxplots inside one plot:

```
#create boxplot that displays temperature distribution for each month in the dataset
colors <- c("#106c00", "#b153bb", "#2c76c4", "#fab923", "#f52929")
months <- unique(airquality$Month)
boxplot(Temp~Month,
data = airquality,
main = "Temperature Distribution by Month",
ylab = "Month",
xlab = "Degrees (F)",
col = colors,
border = "black",
horizontal = TRUE
)
legend("left", legend = months, fill = colors)
```



You can also do this with `ggplot2`:

```
#create horizontal boxplot for points
ggplot(airquality, aes(x = Month, y = Temp, fill = factor(Month), group = Month)) +
  ggtitle("Temperature Distribution by Month") +
  xlab("Months") +
  ylab("Degrees (F)") +
  geom_boxplot() +
  coord_flip() + # this is the argument that flips the boxplots to be horizontal
  #this last line makes the title header bold and in the middle
  theme(plot.title = element_text(hjust = 0.5, size = 16, face = 'bold'),
    legend.position = "bottom")
```


Temperature Distribution by Month
9
8
7
6
5
Months
60
70
80
90
Degrees (F)
factor(Month)
5
6
7
8
9