



CLEANING DATA IN R

Type conversions

Types of variables in R

- character: `"treatment"`, `"123"`, `"A"`
- numeric: `23.44`, `120`, `NaN`, `Inf`
- integer: `4L`, `1123L`
- factor: `factor("Hello")`, `factor(8)`
- logical: `TRUE`, `FALSE`, `NA`

Types of variables in R

```
> class("hello")  
[1] "character"  
  
> class(3.844)  
[1] "numeric"  
  
> class(77L)  
[1] "integer"  
  
> class(factor("yes"))  
[1] "factor"  
  
> class(TRUE)  
[1] "logical"
```

Type conversions

```
> as.character(2016)
[1] "2016"
```

```
> as.numeric(TRUE)
[1] 1
```

```
> as.integer(99)
[1] 99
```

```
> as.factor("something")
[1] something
Levels: something
```

```
> as.logical(0)
[1] FALSE
```

Overview of lubridate

- Written by Garrett Grolmund & Hadley Wickham
- Coerce strings to dates

Dates with lubridate

```
# Load the lubridate package
> library(lubridate)

# Experiment with basic lubridate functions
> ymd("2015-08-25")
[1] "2015-08-25 UTC"      year-month-day

> ymd("2015 August 25")
[1] "2015-08-25 UTC"      year-month-day

> mdy("August 25, 2015")
[1] "2015-08-25 UTC"      month-day-year

> hms("13:33:09")
[1] "13H 33M 9S"          hour-minute-second

> ymd_hms("2015/08/25 13.33.09")
[1] "2015-08-25 13:33:09 UTC" year-month-day hour-minute-second
```



CLEANING DATA IN R

Let's practice!



CLEANING DATA IN R

String manipulation

Overview of stringr

- R package written by Hadley Wickham
- Suite of helpful functions for working with strings
- Functions share consistent interface

Key functions in stringr for cleaning data

```
# Trim leading and trailing white space
> str_trim("  this is a test  ")
[1] "this is a test"      white space removed

# Pad string with zeros
> str_pad("24493", width = 7, side = "left", pad = "0")
[1] "0024493"    7 digits

# Create character vector of names
> friends <- c("Sarah", "Tom", "Alice")

# Search for string in vector
> str_detect(friends, "Alice")
[1] FALSE FALSE  TRUE

# Replace string in vector
> str_replace(friends, "Alice", "David")
[1] "Sarah" "Tom"   "David"
```

Key functions in `stringr` for cleaning data

- `str_trim()` - Trim leading and trailing white space
- `str_pad()` - Pad with additional characters
- `str_detect()` - Detect a pattern
- `str_replace()` - Find and replace a pattern

Other helpful functions in base R

- `tolower()` - Make all lowercase
- `toupper()` - Make all uppercase

```
# Make all lowercase  
> tolower("I AM TALKING LOUDLY!!")  
[1] "i am talking loudly!!"
```

```
# Make all uppercase  
> toupper("I am whispering...")  
[1] "I AM WHISPERING..."
```



CLEANING DATA IN R

Let's practice!



CLEANING DATA IN R

Missing and special values

Missing values

- May be random, but dangerous to assume
- Sometimes associated with variable/outcome of interest
- In R, represented as NA
- May appear in other forms
 - #N/A (Excel)
 - Single dot (SPSS, SAS)
 - Empty string

Special values

- Inf - "Infinite value" (indicative of outliers?)
 - $1/0$
 - $1/0 + 1/0$
 - 33333^{33333}
- NaN - "Not a number" (rethink a variable?)
 - $0/0$
 - $1/0 - 1/0$

Finding missing values

```
# Create small dataset
> df <- data.frame(A = c(1, NA, 8, NA),
                   B = c(3, NA, 88, 23),
                   C = c(2, 45, 3, 1))
```

4 rows, 3 columns

```
# Check for NAs
```

```
> is.na(df)
```

	A	B	C
[1,]	FALSE	FALSE	FALSE
[2,]	TRUE	TRUE	FALSE
[3,]	FALSE	FALSE	FALSE
[4,]	TRUE	FALSE	FALSE

Same size: 4 rows, 3 columns

```
# Are there any NAs?
```

```
> any(is.na(df))
```

```
[1] TRUE
```

```
# Count number of NAs
```

```
> sum(is.na(df))
```

```
[1] 3
```

Finding missing values

```
# Use summary() to find NAs
> summary(df)
```

A		B		C	
Min.	:1.00	Min.	: 3.0	Min.	: 1.00
1st Qu.:	2.75	1st Qu.:	13.0	1st Qu.:	1.75
Median	:4.50	Median	:23.0	Median	: 2.50
Mean	:4.50	Mean	:38.0	Mean	:12.75
3rd Qu.:	6.25	3rd Qu.:	55.5	3rd Qu.:	13.50
Max.	:8.00	Max.	:88.0	Max.	:45.00
NA's	:2	NA's	:1		

Dealing with missing values

```
# Find rows with no missing values
> complete.cases(df)
[1] TRUE FALSE TRUE FALSE

# Subset data, keeping only complete cases
> df[complete.cases(df), ]
  A  B C
1 1  3 2
3 8 88 3

# Another way to remove rows with NAs
> na.omit(df)
  A  B C
1 1  3 2
3 8 88 3
```



CLEANING DATA IN R

Let's practice!

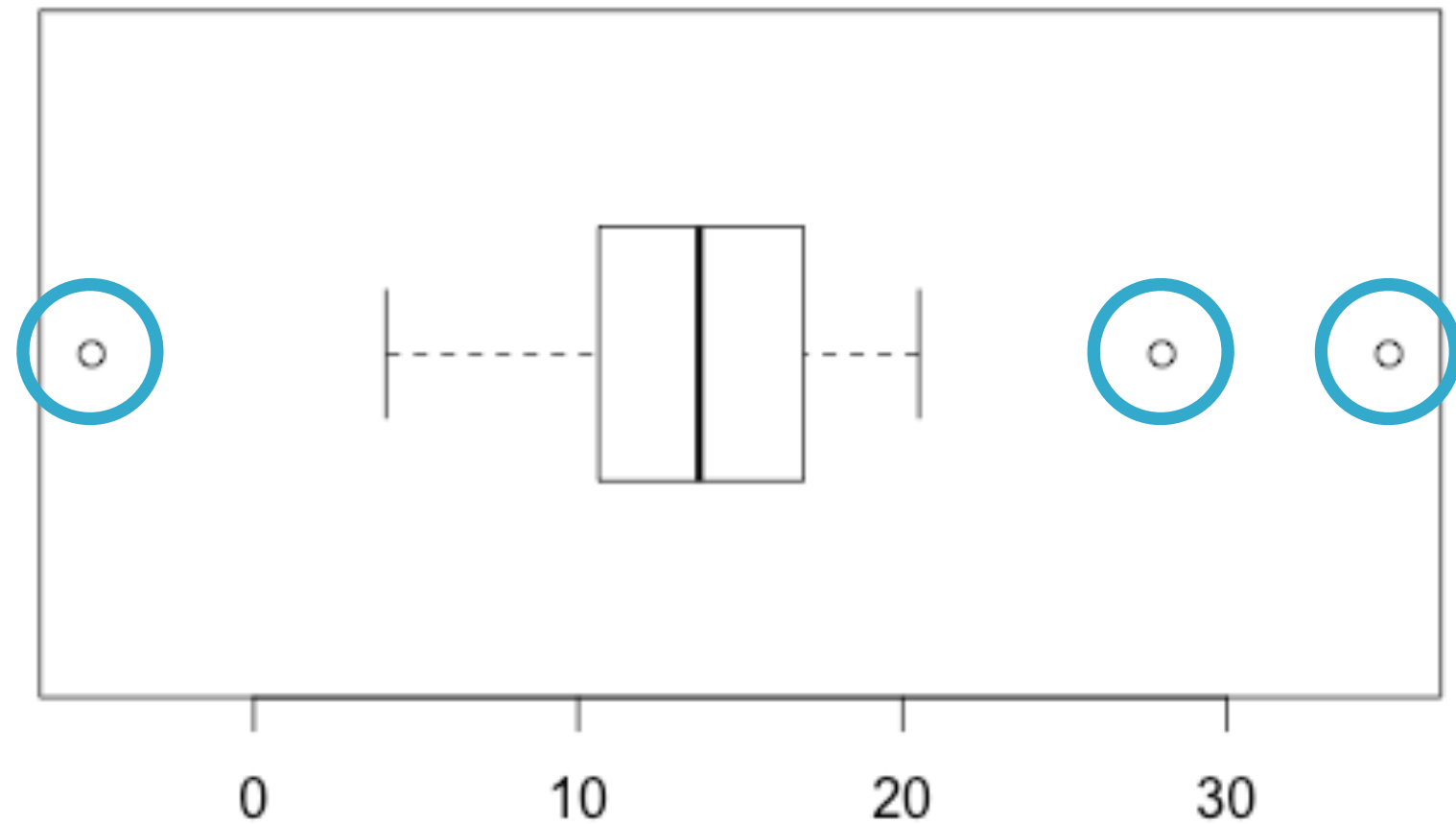


CLEANING DATA IN R

Outliers and obvious errors

Outliers

```
# Simulate some data  
> set.seed(10)  
> x <- c(rnorm(30, mean = 15, sd = 5), -5, 28, 35)  
  
# View a boxplot  
> boxplot(x, horizontal = TRUE)
```



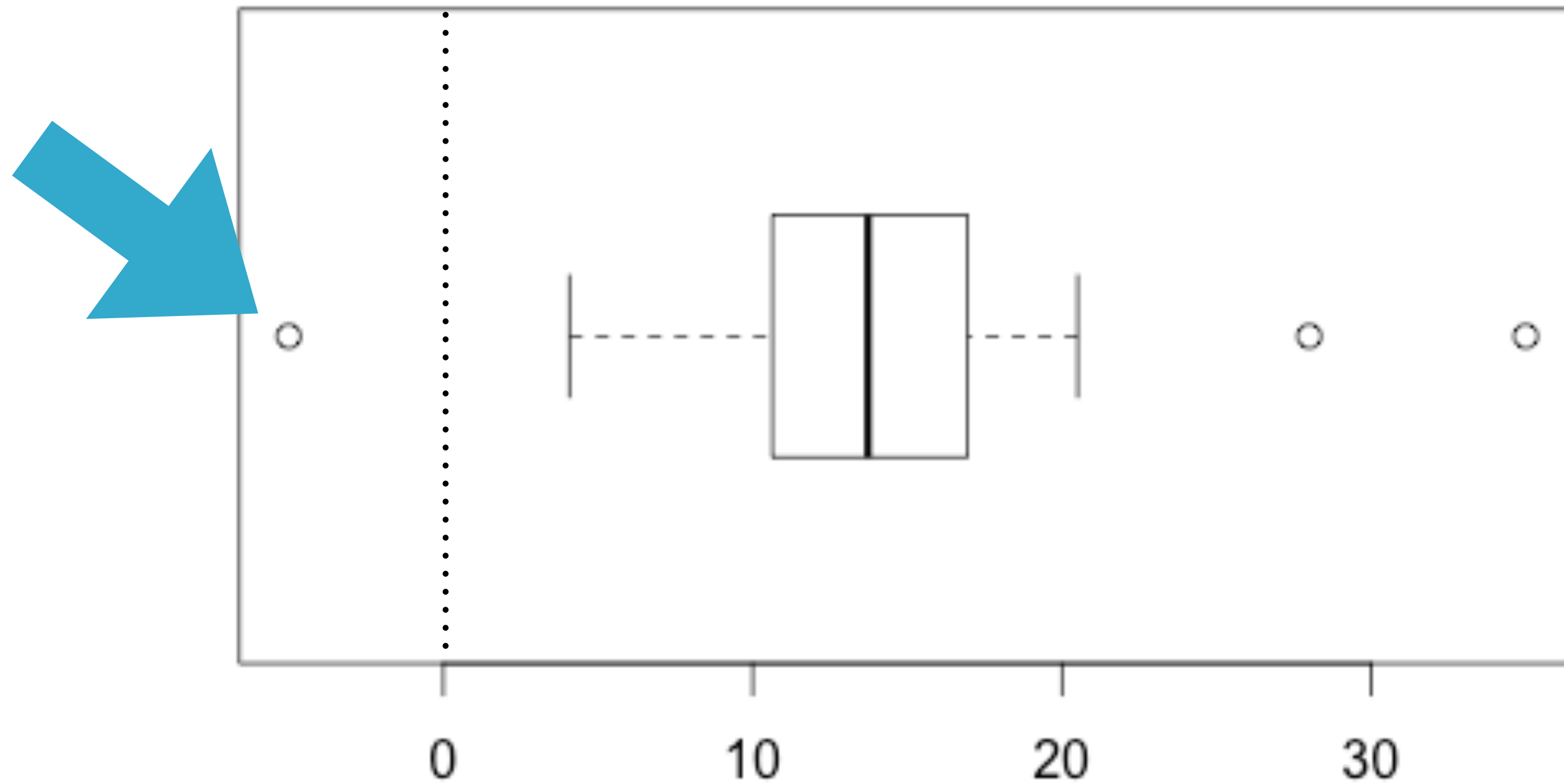
Outliers

Outliers

- Extreme values distant from other values
- Several causes
 - Valid measurements
 - Variability in measurement
 - Experimental error
 - Data entry error
- May be discarded or retained depending on cause

Obvious errors

What if these values are supposed to represent ages?



Obvious errors

- May appear in many forms
 - Values so extreme they can't be plausible (e.g. person aged 243)
 - Values that don't make sense (e.g. negative age)
- Several causes
 - Measurement error
 - Data entry error
 - Special code for missing data (e.g. -1 means missing)
- Should generally be removed or replaced

Finding outliers and errors

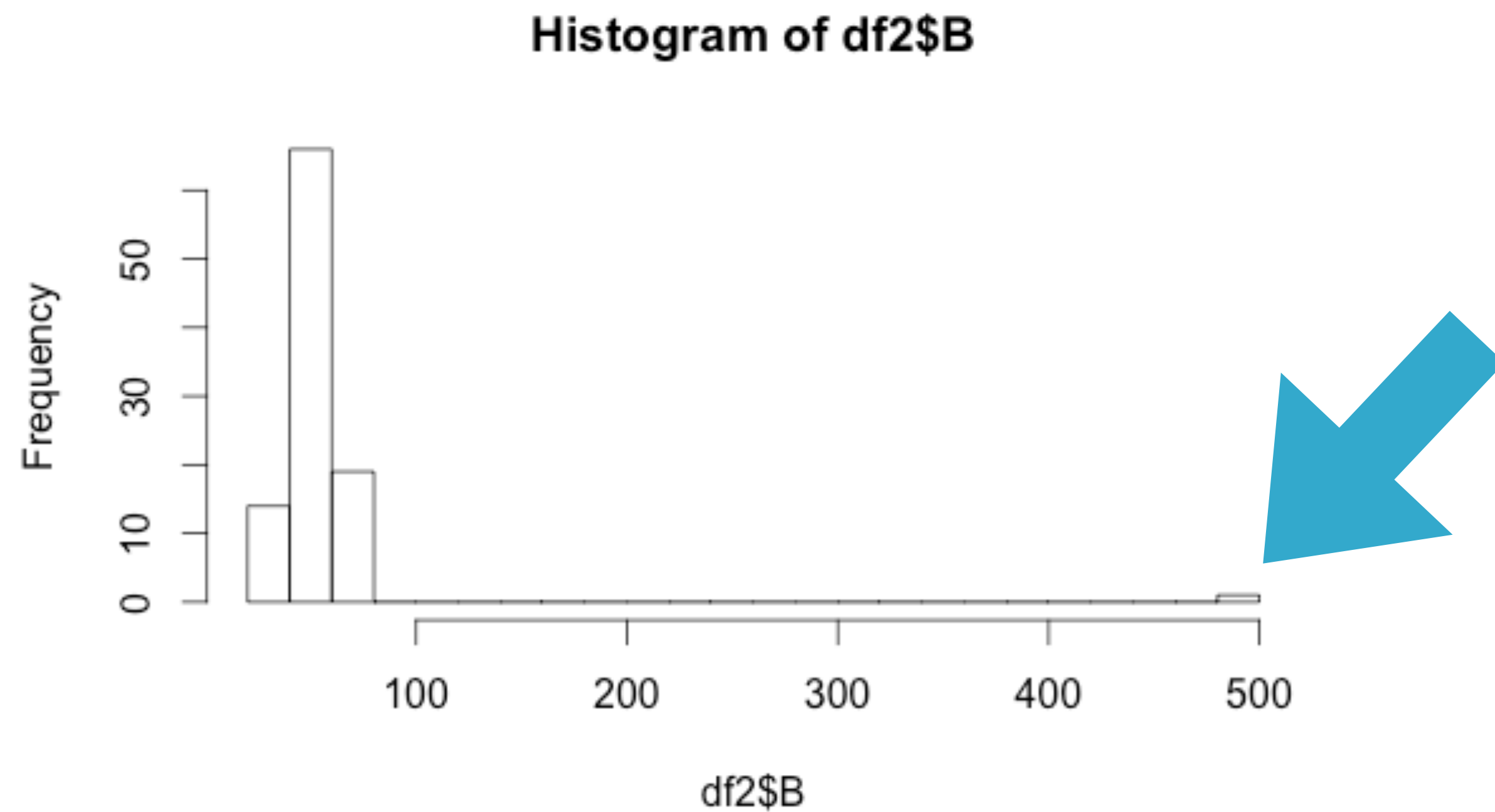
```
# Create another small dataset
> df2 <- data.frame(A = rnorm(100, 50, 10),
                    B = c(rnorm(99, 50, 10), 500),
                    C = c(rnorm(99, 50, 10), -1))
```

```
# View a summary
> summary(df2)
```

A		B		C	
Min.	:23.7	Min.	: 26.9	Min.	:-1.0
1st Qu.	:43.7	1st Qu.	: 43.7	1st Qu.	:40.3
Median	:51.9	Median	: 49.8	Median	:48.5
Mean	:50.4	Mean	: 54.9	Mean	:47.8
3rd Qu.	:56.9	3rd Qu.	: 56.6	3rd Qu.	:56.3
Max.	:77.2	Max.	:500.0	Max.	:75.1

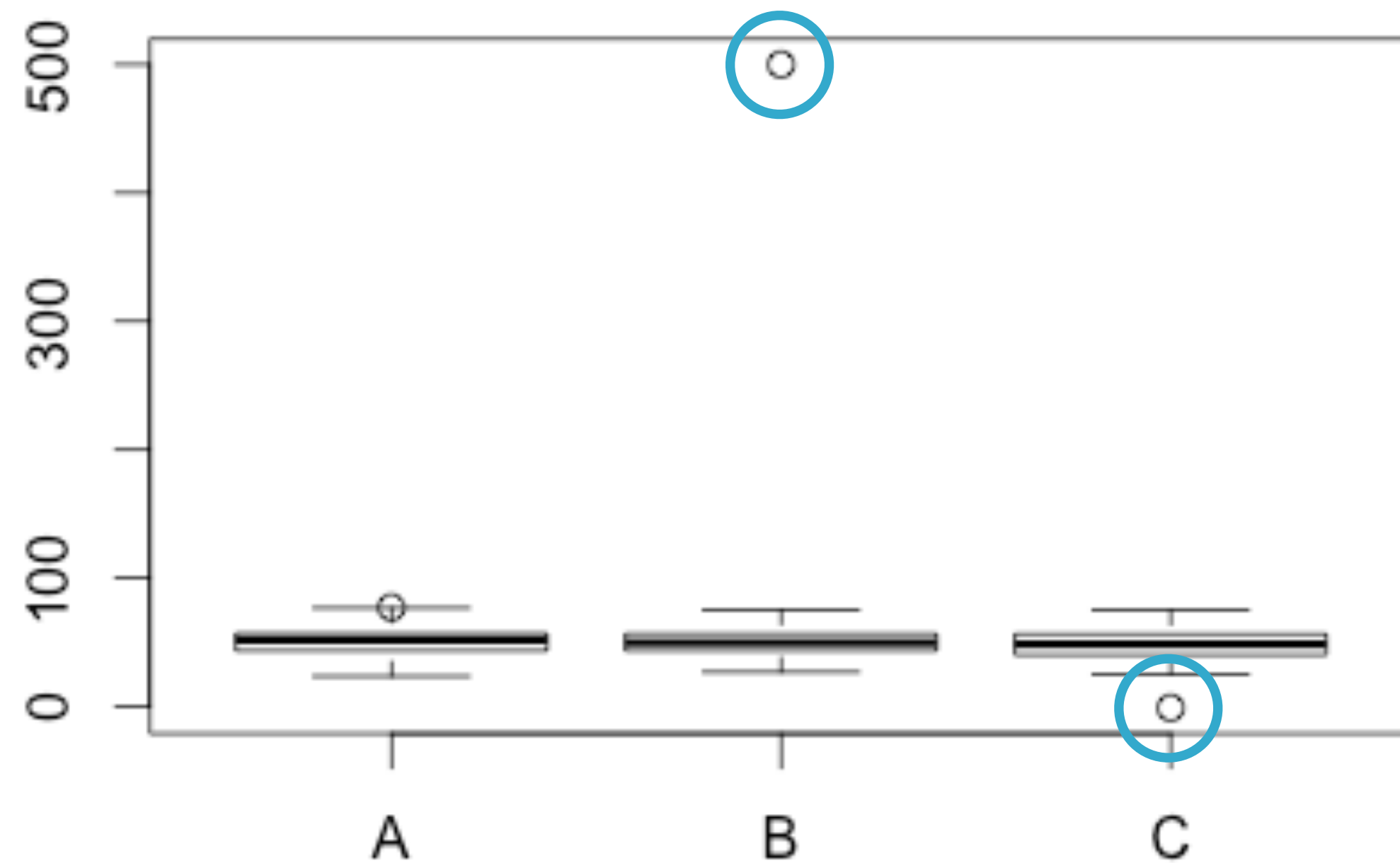
Finding outliers and errors

```
# View a histogram  
> hist(df2$B, breaks = 20)
```



Finding outliers and errors

```
# View a boxplot  
> boxplot(df2)
```





CLEANING DATA IN R

Let's practice!