

# Statistical tests

Day 3 - Introduction to Data Analysis with R

Selina Baldauf

Freie Universität Berlin - Theoretical Ecology

September 19, 2024

# Overview of tests



Please note the discussion going on about p-values as a basis for binary decisions. See e.g. [here](#) or [here](#) as a starting

# Tests for normal distribution

# Test for normal distribution

There are **various tests** and the outcome might differ!

## Shapiro-Wilk-Test

- How much does variance of observed data differ from normal distribution
- Specific test only for normal distribution
- High power, also for few data points

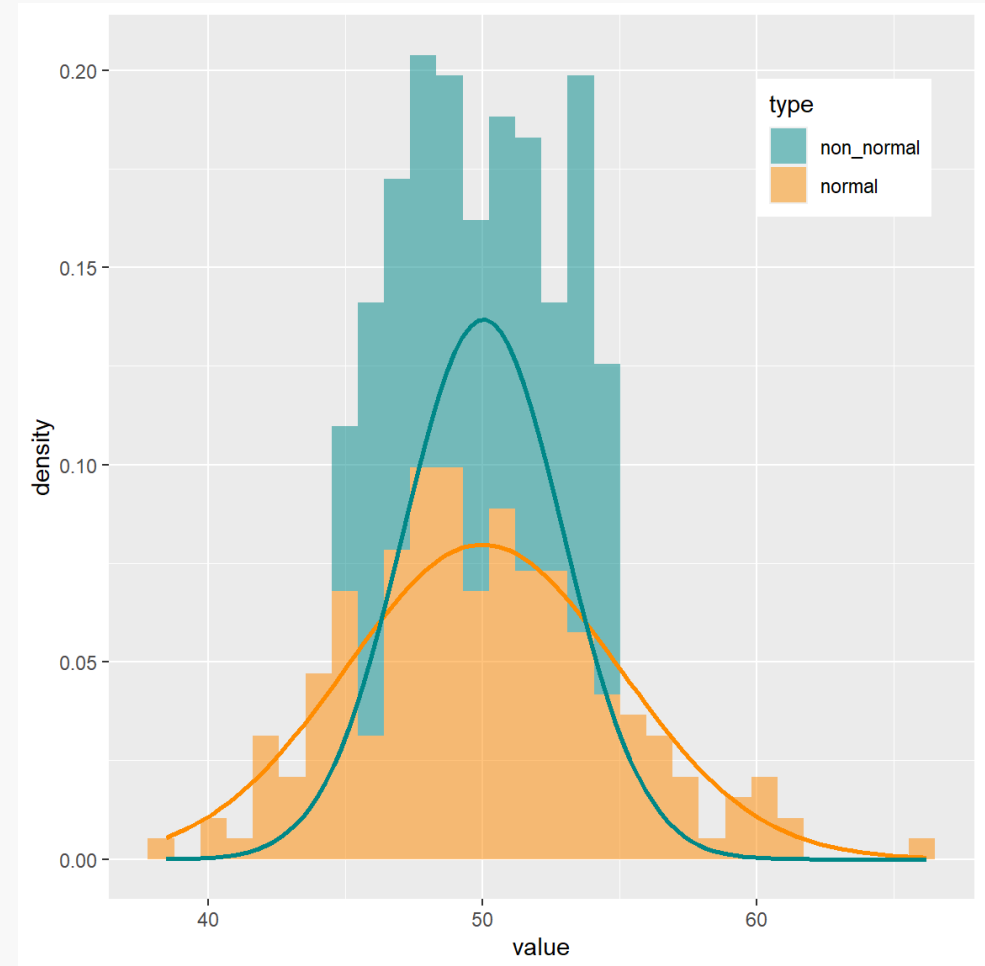
## Visual tests: QQ-Plot

- Quantiles of observed data plotted against quantiles of normal distribution
- Scientist has to decide if normal or not

# The data

A tibble with two variables

- **normal1**: 200 normally distributed values with mean 50 and standard deviation 5
- **non\_normal1**: 200 uniformly distributed values between 45 and 55



# Shapiro-Wilk-Test

$H_0$ : Data does not differ from a normal distribution

```
shapiro.test(mydata$normal)
#>
#>  Shapiro-Wilk normality test
#>
#> data:  mydata$normal
#> W = 0.99076, p-value = 0.2298
```

- W: test statistic
- p-value: probability to observe the data if was true

The data does not deviate significantly from a normal distribution (Shapiro-Wilk-Test,  $W = 0.991$ ,  $p = 0.23$ ).

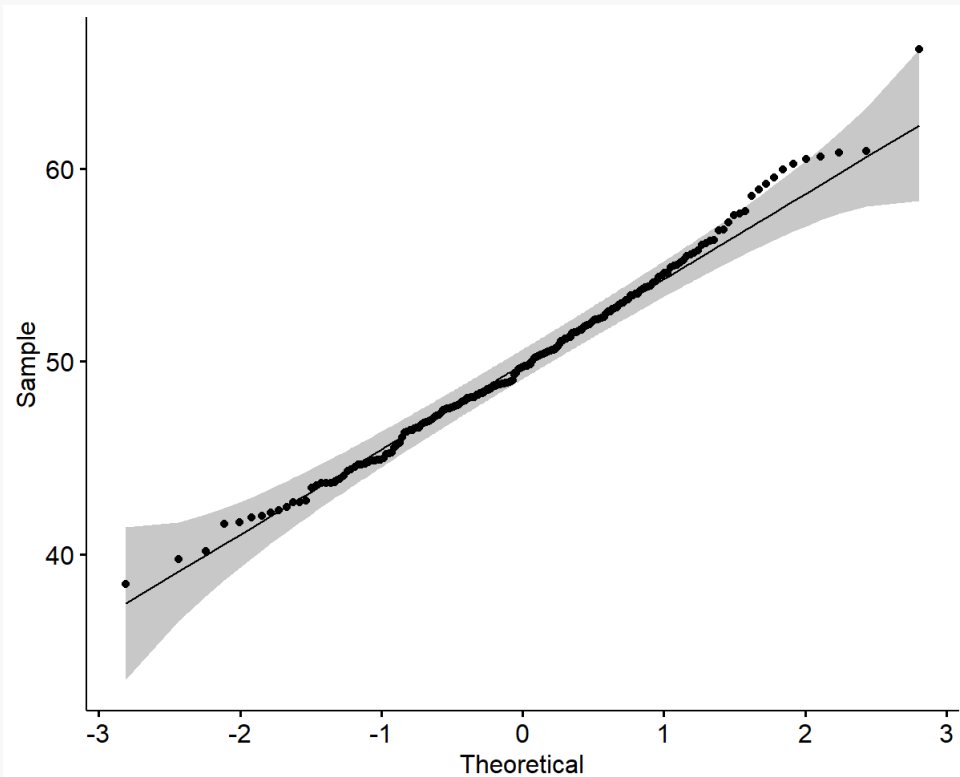
```
shapiro.test(mydata$non_normal)
#>
#>  Shapiro-Wilk normality test
#>
#> data:  mydata$non_normal
#> W = 0.95114, p-value = 2.435e-06
```

The data deviates significantly from a normal distribution (Shapiro-Wilk-Test,  $W = 0.95$ ,  $p < 0.001$ ).

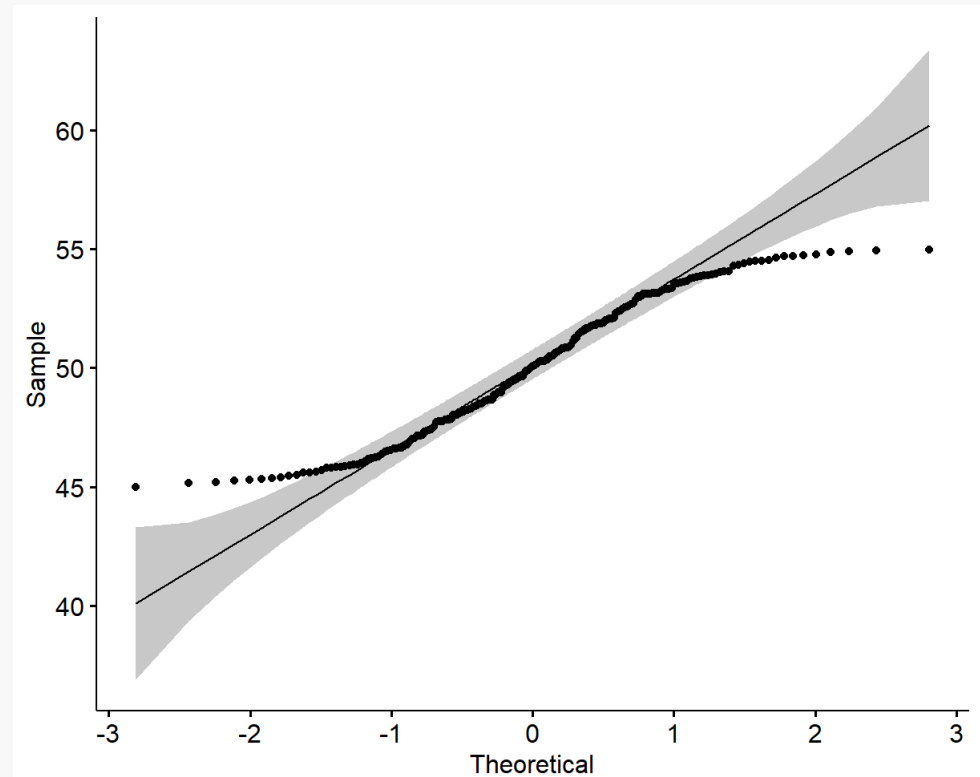
# Visual test with QQ-Plot

Points should match the straight line. Small deviations are okay.

```
# ggplot(mydata, aes(sample = normal))  
#   stat_qq() + stat_qq_line()  
ggpubr::ggqqplot(mydata$normal)
```



```
# ggplot(mydata, aes(sample = non_normal))  
#   stat_qq() + stat_qq_line()  
ggpubr::ggqqplot(mydata$non_normal)
```



# Tests for equal variance



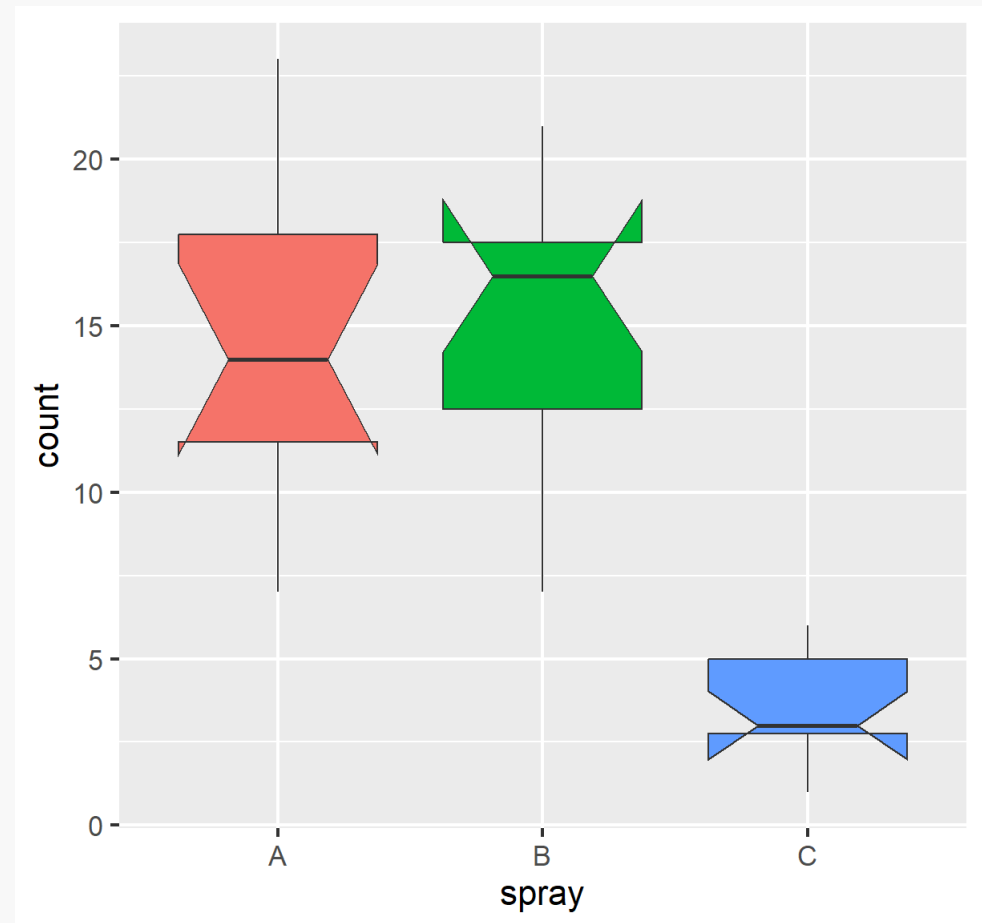
# The data

Counts of insects in agricultural units treated with different insecticides.

Compare treatments A, B and C:

Create subsets before: count variable for each treatment as a vector

```
TreatA <- filter(  
  InsectSprays,  
  spray == "A")$count  
TreatB <- filter(  
  InsectSprays,  
  spray == "B")$count  
TreatC <- filter(  
  InsectSprays,  
  spray == "C")$count
```



# Test for equal variance

First, test for normal distribution!

## F-Test

- Normal distribution of groups
- Calculates ratio of variances (if equal, ratio = 1)
- p: How likely is ratio if variances were equal?

## Levene test

- Non-normal distribution of groups
- Compare difference between data sets with difference within data sets

# Test for equal variance

First, test for normal distribution

```
shapiro.test(TreatA)
#>
#>  Shapiro-Wilk normality test
#>
#> data:  TreatA
#> W = 0.95757, p-value = 0.7487
shapiro.test(TreatB)
#>
#>  Shapiro-Wilk normality test
#>
#> data:  TreatB
#> W = 0.95031, p-value = 0.6415
shapiro.test(TreatC)
#>
#>  Shapiro-Wilk normality test
#>
#> data:  TreatC
#> W = 0.92128, p-value = 0.2967
```

Result: All 3 treatments are normally distributed.

# F-Test

: Variances do not differ between groups

```
var.test(TreatA, TreatB)
#>
#> F test to compare two variances
#>
#> data: TreatA and TreatB
#> F = 1.2209, num df = 11, denom df = 11, p-value = 0.7464
#> alternative hypothesis: true ratio of variances is not equal to 1
#> 95 percent confidence interval:
#> 0.3514784 4.2411442
#> sample estimates:
#> ratio of variances
#> 1.22093
```

- F: test statistics, ratio of variances (if  $F = 1$ , variances are equal)
- df: degrees of freedom of both groups
- p-value: how likely is it to observe the data if was true?

Variances of sprays A & B don't differ significantly (F-Test,  $= 1.22$ ,  $p = 0.75$ )

# F-Test

: Variances do not differ between groups

```
var.test(TreatA, TreatC)
#>
#>  F test to compare two variances
#>
#> data:  TreatA and TreatC
#> F = 7.4242, num df = 11, denom df = 11, p-value = 0.002435
#> alternative hypothesis: true ratio of variances is not equal to 1
#> 95 percent confidence interval:
#>    2.137273 25.789584
#> sample estimates:
#> ratio of variances
#>                7.424242
```

Variances of sprays A & C differ significantly (F-Test, = 7.42,  $p = 0.002$ )

# Test for equal means

# Test for equal means

## t-test

- Normal distribution AND equal variance
- Compares if mean values are within range of standard error of each other
- $p$ : how likely is the difference if the means were equal

## Welch-Test

- Normal distribution but unequal variance
- Corrected t-test

## Wilcoxon rank sum test

- Non-normal distribution and unequal variance
- Compares rank sums of the data
- Non-parametric

# t-test

: The samples do not differ in their mean

Treatment A and B: **normally distributed** and **equal variance**

```
t.test(TreatA, TreatB, var.equal = TRUE)
#>
#> Two Sample t-test
#>
#> data: TreatA and TreatB
#> t = -0.45352, df = 22, p-value = 0.6546
#> alternative hypothesis: true difference in means is not equal to 0
#> 95 percent confidence interval:
#> -4.643994 2.977327
#> sample estimates:
#> mean of x mean of y
#> 14.50000 15.33333
```

- t: test statistics ( $t = 0$  means equal means)
- df: degrees of freedom of t-statistics
- p-value: how likely is it to observe the data if was true?



# t-test

: The samples do not differ in their mean

Treatment A and B: **normally distributed** and **equal variance**

```
t.test(TreatA, TreatB, var.equal = TRUE)
#>
#> Two Sample t-test
#>
#> data: TreatA and TreatB
#> t = -0.45352, df = 22, p-value = 0.6546
#> alternative hypothesis: true difference in means is not equal to 0
#> 95 percent confidence interval:
#> -4.643994 2.977327
#> sample estimates:
#> mean of x mean of y
#> 14.50000 15.33333
```

**Result:** The means of spray A and B don't differ significantly ( $t = -0.45$ ,  $df = 22$ ,  $p = 0.66$ )

# Welch-Test

: The samples do not differ in their mean

Treatment A and C: **normally distributed** and **non-equal variance**

```
t.test(TreatA, TreatC, var.equal = FALSE)
#>
#> Welch Two Sample t-test
#>
#> data: TreatA and TreatC
#> t = 7.5798, df = 13.91, p-value = 2.655e-06
#> alternative hypothesis: true difference in means is not equal to 0
#> 95 percent confidence interval:
#>  7.885546 14.114454
#> sample estimates:
#> mean of x mean of y
#>      14.5      3.5
```

**Result:** The means of spray A and C do differ significantly ( $t = 7.58$ ,  $df = 13.9$ ,  $p < 0.001$ )

# Wilcoxon-rank-sum Test

: The samples do not differ in their mean

We don't need the Wilcoxon test to compare treatment A and B, but for the sake of an example:

```
wilcox.test(TreatA, TreatB)
#>
#>  Wilcoxon rank sum test with continuity correction
#>
#> data:  TreatA and TreatB
#> W = 62, p-value = 0.5812
#> alternative hypothesis: true location shift is not equal to 0
```

**Result:** The means of spray A and B do not differ significantly ( $W = 62$ ,  $p = 0.58$ )

# Paired values

Are there pairs of data points?

**Example:** samples of invertebrates across various rivers before and after sewage plants.

- For each plant, there is a pair of data points (before and after the plant)
- Question: Is the change (before-after) significant

Use **paired = TRUE** in the test.

```
t.test(TreatA, TreatB, var.equal = TRUE, paired = TRUE)
t.test(TreatA, TreatB, var.equal = FALSE, paired = TRUE)
wilcox.test(TreatA, TreatB, paired = TRUE)
```

Careful: your treatment vector both have to have the same order

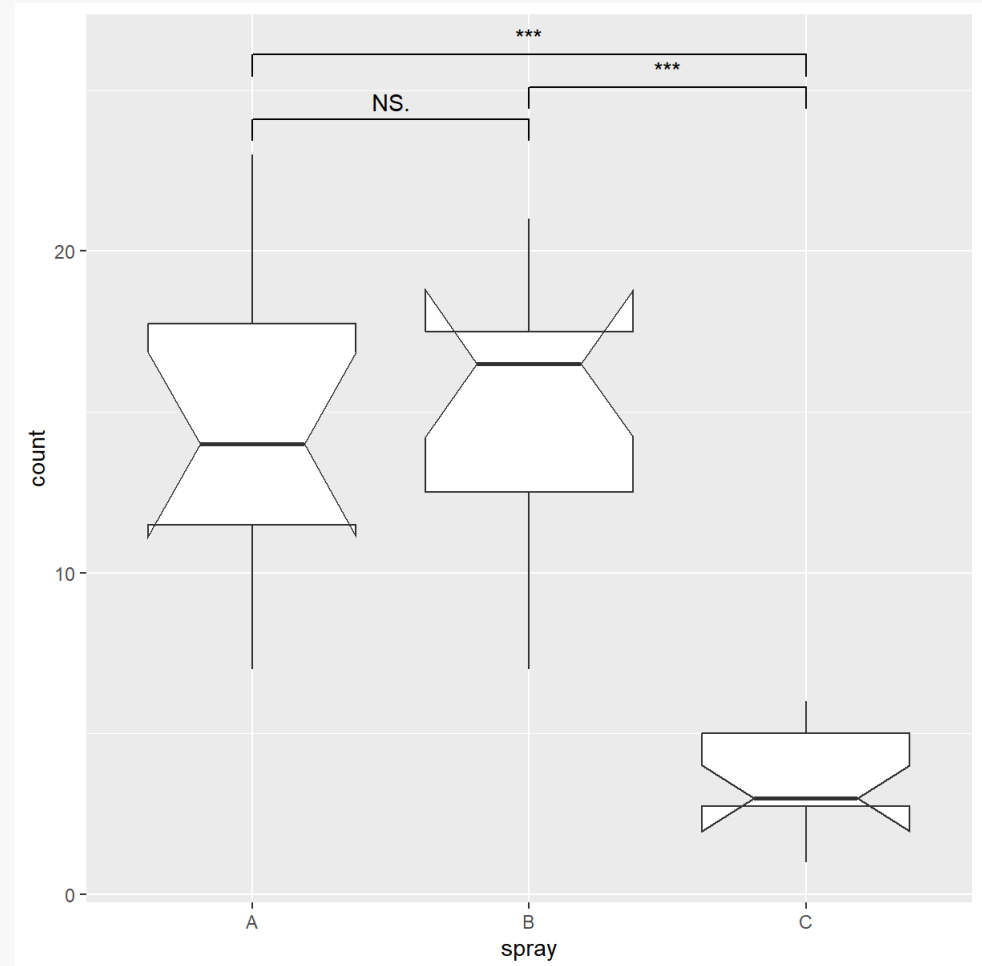
# Plot test results with **ggsignif**

The **ggsignif** package offers a **geom\_signif()** layer that can be added to a ggplot to annotate significance levels

```
# install.packages("ggsignif")  
library(ggsignif)
```

# Plot test results with `geom_signif()`

```
1 ggplot(  
2   InsectSprays,  
3   aes(x = spray, y = count)  
4 ) +  
5   geom_boxplot(notch = TRUE) +  
6   geom_signif(  
7     comparisons = list(  
8       c("A", "B"),  
9       c("B", "C"),  
10      c("A", "C")  
11    ),  
12    map_signif_level = TRUE,  
13    y_position = c(23, 24, 25)  
14  )
```



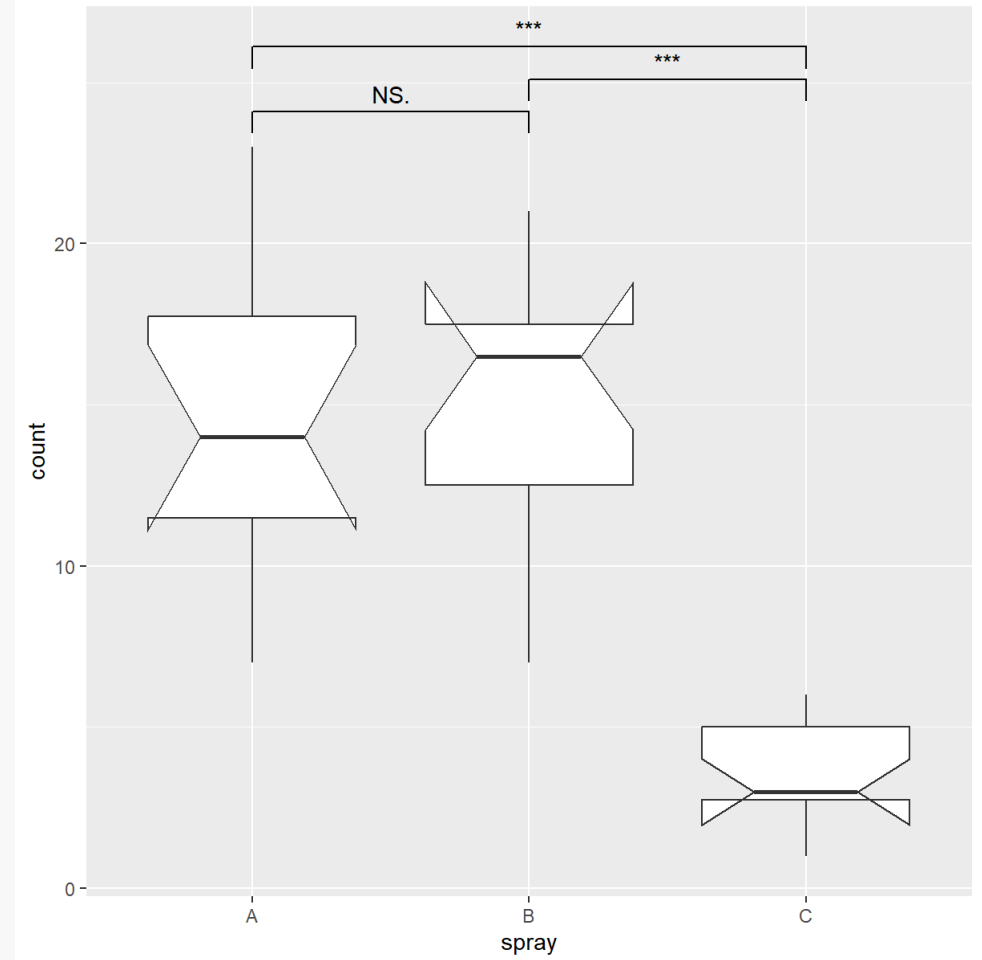
- By default, a Wilcoxon test is performed

# Plot test results with `geom_signif()`

```

1  ggplot(
2    InsectSprays,
3    aes(x = spray, y = count)
4  ) +
5    geom_boxplot(notch = TRUE) +
6    geom_signif(
7      comparisons = list(
8        c("A", "B"),
9        c("B", "C"),
10       c("A", "C")
11      ),
12      test = "t.test",
13      test.args = list(
14        var.equal = TRUE
15      ),
16      map_signif_level = TRUE,
17      y_position = c(23, 24, 25)
18    )

```



- **test**: run specific test
- **test.args**: pass additional arguments in a list
- **?geom\_signif** for more options

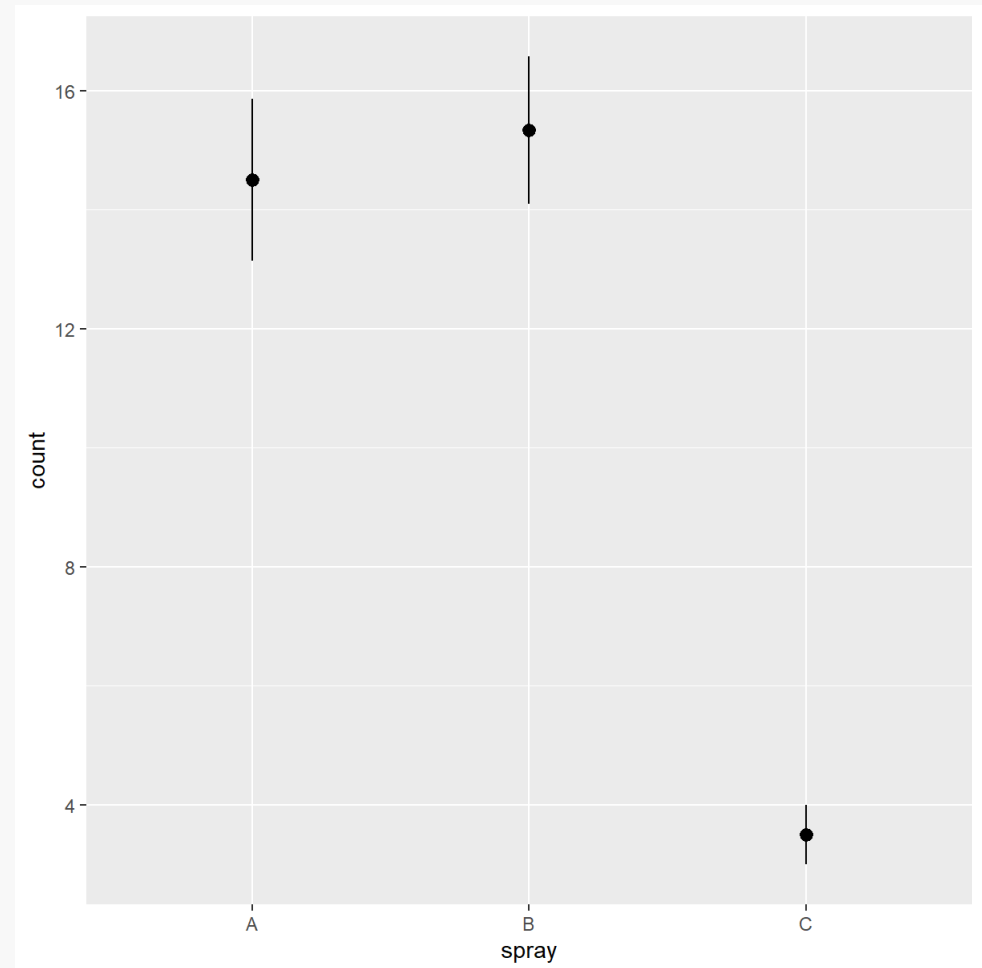


# Plot mean $\pm$ se using `stat_summary`

Another way to plot the results is to plot mean and standard error of the mean:

```
1 ggplot(  
2   InsectSprays,  
3   aes(x = spray, y = count)  
4 ) +  
5   stat_summary()
```

- By default `stat_summary` adds mean and standard error of the mean as pointrange

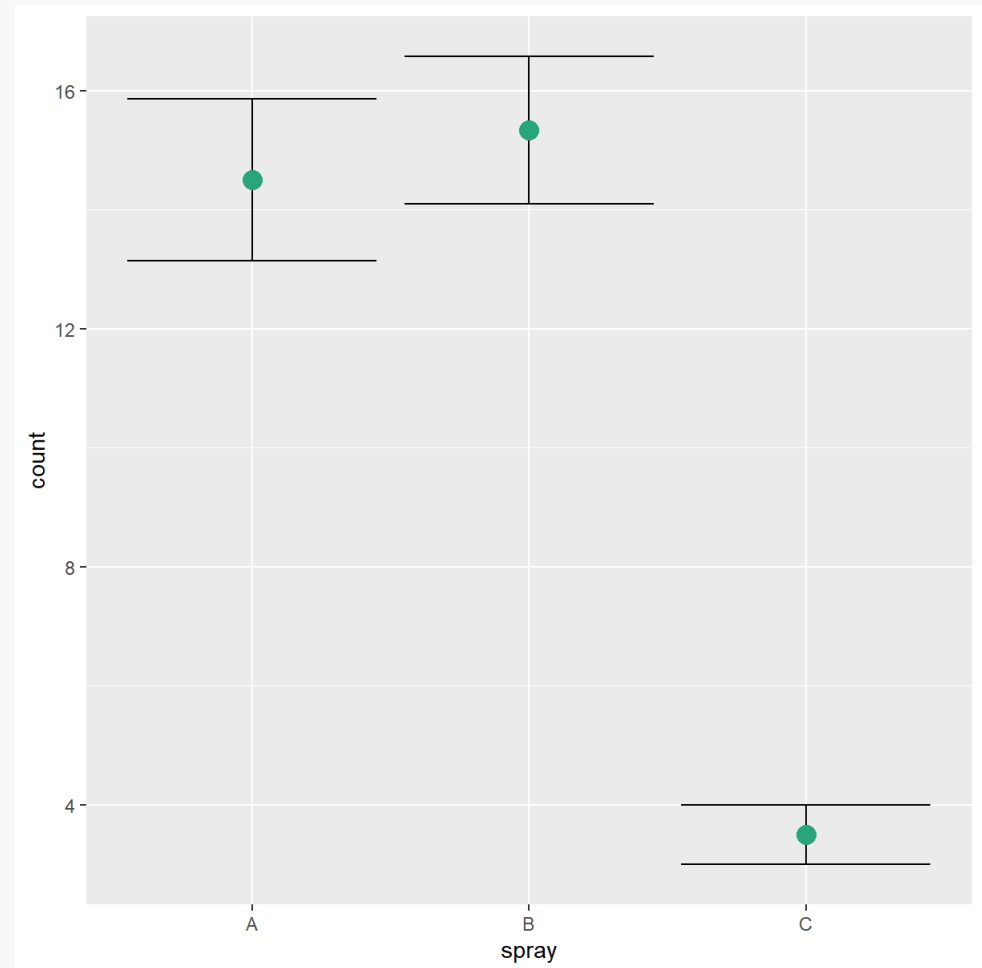


# Plot mean $\pm$ se using `stat_summary`

Another way to plot the results is to plot mean and standard error of the mean:

```
1 ggplot(  
2   InsectSprays,  
3   aes(x = spray, y = count)  
4 ) +  
5   stat_summary(  
6     fun.data = mean_se,  
7     geom = "errorbar"  
8 ) +  
9   stat_summary(  
10    fun.y = mean,  
11    geom = "point",  
12    color = "#28a87d",  
13    size = 4  
14 )
```

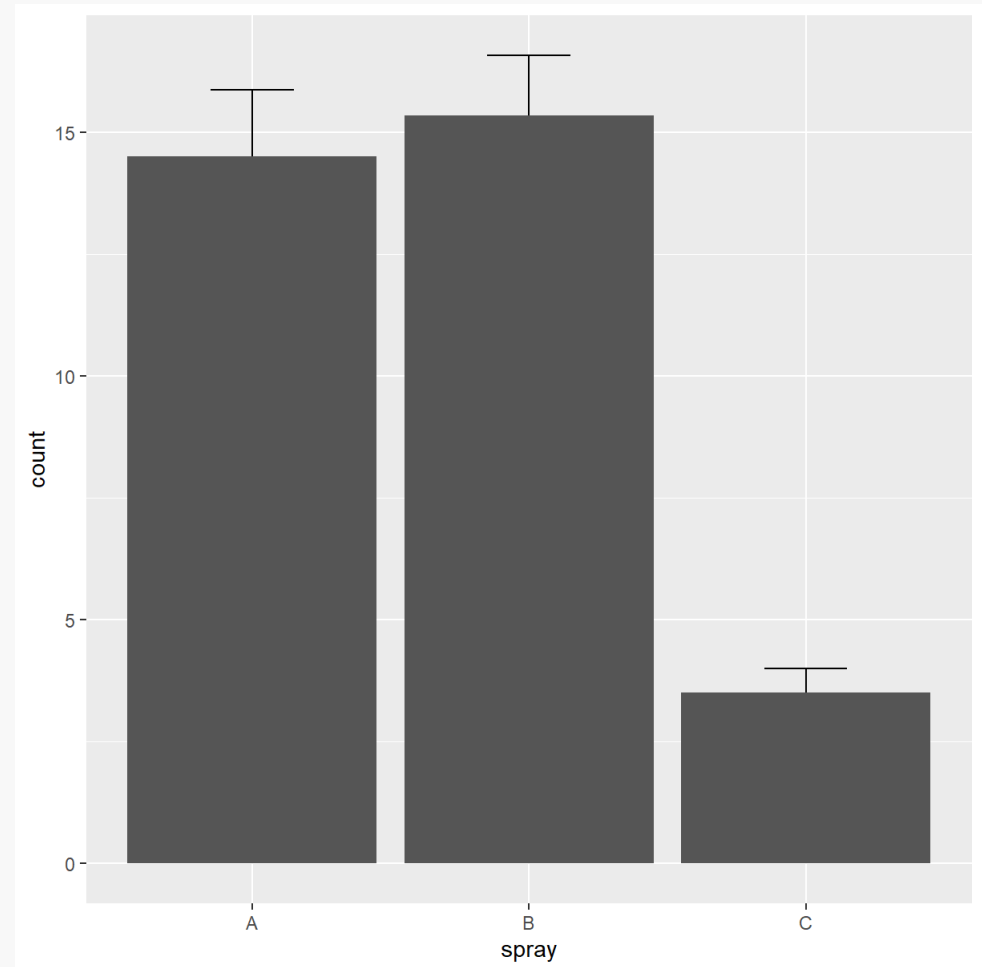
- Inside `stat_summary`, define summary function
  - `fun.data` for errorbars, `fun.y` for points (e.g. mean)



# Plot mean $\pm$ se using `stat_summary`

Another way to plot the results is to plot mean and standard error of the mean:

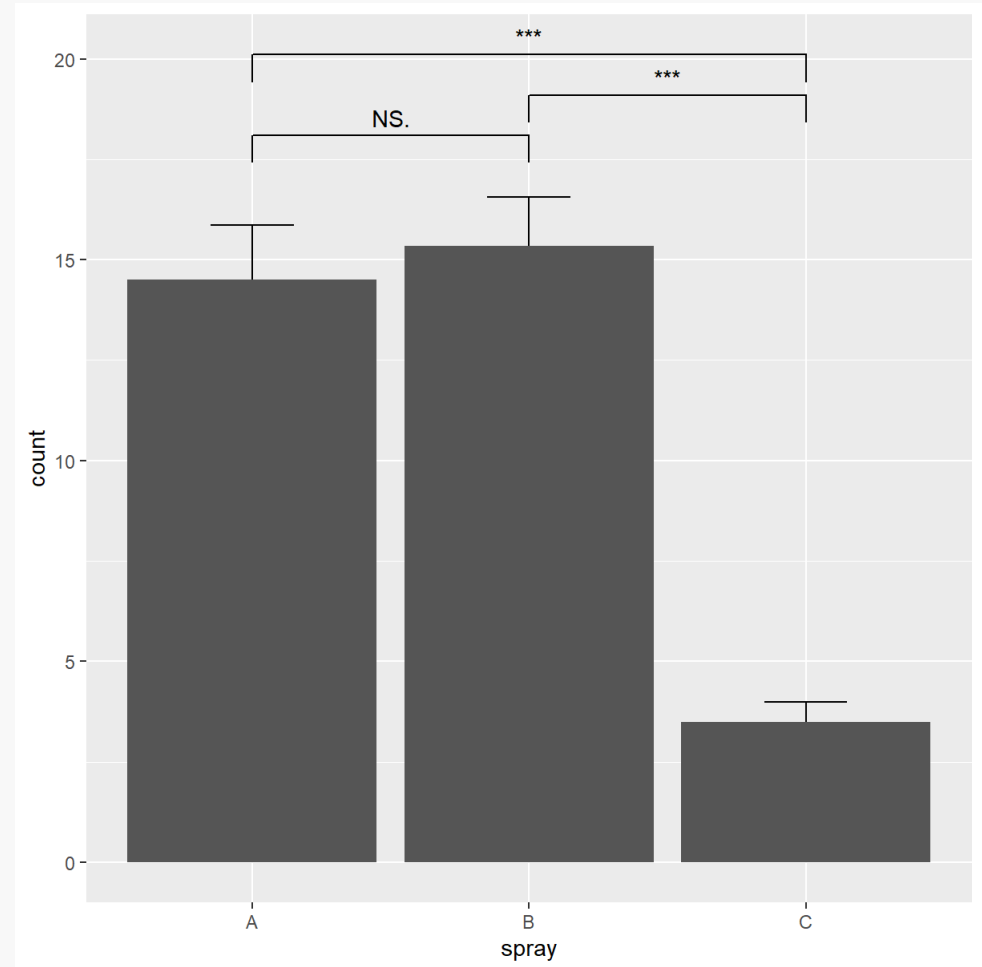
```
1 ggplot(  
2   InsectSprays,  
3   aes(x = spray, y = count)  
4 ) +  
5   stat_summary(  
6     fun.data = mean_se,  
7     geom = "errorbar",  
8     width = 0.3  
9   ) +  
10  stat_summary(  
11    fun.y = mean,  
12    geom = "bar",  
13    size = 4  
14  )
```



# Plot mean $\pm$ se using `stat_summary`

Just like before, you can also add a `geom_signif` to a barplot:

```
1 ggplot(  
2   InsectSprays,  
3   aes(x = spray, y = count)  
4 ) +  
5   stat_summary(  
6     fun.data = mean_se,  
7     geom = "errorbar",  
8     width = 0.3  
9   ) +  
10  stat_summary(  
11    fun.y = mean,  
12    geom = "bar"  
13  ) +  
14  ggsignif::geom_signif(  
15    comparisons = list(  
16      c("A", "B"),  
17      c("B", "C"),  
18      c("A", "C")  
19    ),
```



# Now you

Task 1 (45) min)

Statistical tests

Find the task description [here](#)

