

# R Workshop: Correlation

Brier Gallihugh, M.S.

2023-06-19

## Table of contents

Introduction . . . . .	1
Statistical Assumptions . . . . .	2
Normality of Variables . . . . .	2
Graphical Depiction of Normality Assumption (mpg) . . . . .	2
Statistical Depiction of Normality Assumption (mpg) . . . . .	3
Graphical Depiction of Normality Assumption (wt) . . . . .	3
Statistical Depiction of Normality Assumption (wt) . . . . .	4
Running An Actual Correlation . . . . .	4

## Introduction

Correlations might be the most common statistical test run (especially early on in a project). For this part of the workshop we will be using the `mtcars` data set due to its many numerical values that we can assess for correlation. Below we will see the first 10 rows of the data set displayed

```
library(tidyverse)
library(car)
library(psych)

data <- mtcars

print(head(data,10))
```

①

- ① Display the first 10 rows of the `mtcars` data. You can also return the last 10 rows with the following function `tail(data,10)`

	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
Mazda RX4	21.0	6	160.0	110	3.90	2.620	16.46	0	1	4	4
Mazda RX4 Wag	21.0	6	160.0	110	3.90	2.875	17.02	0	1	4	4
Datsun 710	22.8	4	108.0	93	3.85	2.320	18.61	1	1	4	1
Hornet 4 Drive	21.4	6	258.0	110	3.08	3.215	19.44	1	0	3	1
Hornet Sportabout	18.7	8	360.0	175	3.15	3.440	17.02	0	0	3	2
Valiant	18.1	6	225.0	105	2.76	3.460	20.22	1	0	3	1
Duster 360	14.3	8	360.0	245	3.21	3.570	15.84	0	0	3	4
Merc 240D	24.4	4	146.7	62	3.69	3.190	20.00	1	0	4	2
Merc 230	22.8	4	140.8	95	3.92	3.150	22.90	1	0	4	2
Merc 280	19.2	6	167.6	123	3.92	3.440	18.30	1	0	4	4

## Statistical Assumptions

The primary assumption of a **Pearson's correlation coefficient** is that the data is on some kind of interval scale. However, if we wish to generalize, we must have a random large sample (unlikely) and the individual variables should be roughly normally distributed. This is the assumption we will focus on for this part of the workshop.

## Normality of Variables

For this part of the workshop, we are going to focus on the `mpg` and `wt` variables. We will be looking at normality both statistically<sub>1</sub> as well as graphically.

### ! A Note About Statistical Assumption Testing

1. Odds are your statistical test is going to fail pretty much every single time. Particularly if you use something like a Shapiro Wilk test, but we'll look at it anyway

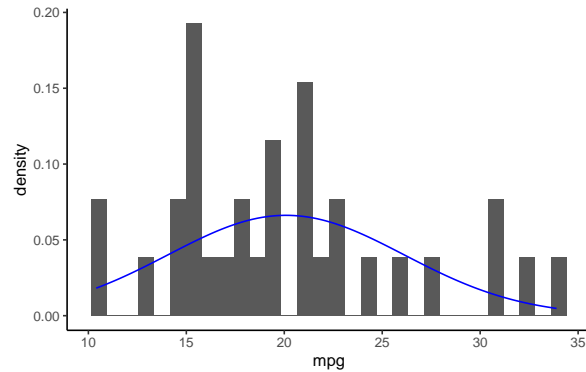
## Graphical Depiction of Normality Assumption (mpg)

```
mpg_plot <- ggplot(data,aes(x = mpg)) +
  geom_histogram(aes(y=after_stat(density))) +
  stat_function(fun = dnorm,
               args = list(mean = mean(data$mpg),
                           sd = sd(data$mpg)),
               col = "blue") +
  theme_classic()

print(mpg_plot)
```

①

- ① This should look very familiar to the histogram part of the workshop



### Statistical Depiction of Normality Assumption (mpg)

```
print(psych::describe(data$mpg))
```

①

```
print(shapiro.test(data$mpg))
```

②

- ① The `psych` package has a bunch of nifty functions for social science research. One is the `describe()` function which gives you a bunch of variable level summary statistics (e.g., mean, median, se, etc.)
- ② The `shapiro.test()` performs a Shapiro Wilk test of normality. Keep in mind this particular test is very sensitive to sample sizes

	vars	n	mean	sd	median	trimmed	mad	min	max	range	skew	kurtosis	se
X1	1	32	20.09	6.03	19.2	19.7	5.41	10.4	33.9	23.5	0.61	-0.37	1.07

Shapiro-Wilk normality test

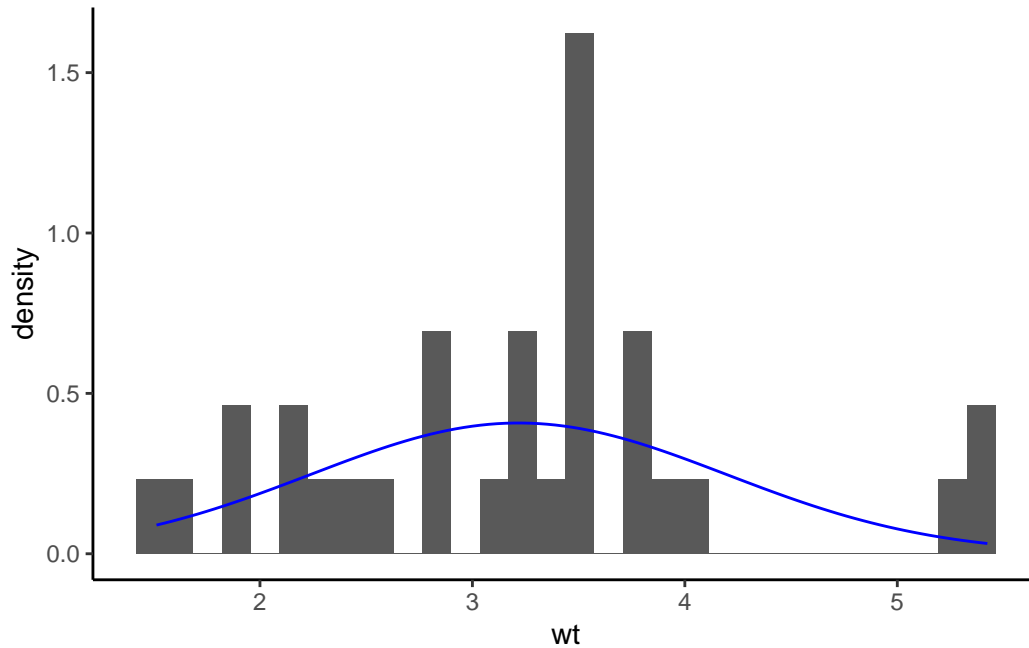
data: data\$mpg

W = 0.94756, p-value = 0.1229

### Graphical Depiction of Normality Assumption (wt)

```
wt_plot <- ggplot(data, aes(x = wt)) +
  geom_histogram(aes(y=after_stat(density))) +
  stat_function(fun = dnorm,
               args = list(mean = mean(data$wt),
                           sd = sd(data$wt)),
               col = "blue") +
  theme_classic()
```

```
print(wt_plot)
```



### Statistical Depiction of Normality Assumption (wt)

```
print(psych::describe(data$wt))  
print(shapiro.test(data$wt))
```

```
vars  n mean  sd median trimmed  mad  min  max range skew kurtosis  se  
X1    1 32 3.22 0.98  3.33   3.15 0.77 1.51 5.42  3.91 0.42   -0.02 0.17  
Shapiro-Wilk normality test
```

```
data: data$wt  
W = 0.94326, p-value = 0.09265
```

### Running An Actual Correlation

There are multiple packages and methods for calculating a correlation in R depending on what you want to assess. The best to use for psychology is probably the `corr.test()` function in the `psych` package because it allows you to change the type of correlation you wish to compute (e.g., spearman vs pearson) as well as generate confidence intervals and do p value adjustments

```
corr_results <- corr.test(x = data$mpg, ①
                        y = data$wt,    ②
                        use = "pairwise", ③
                        method = "pearson", ④
                        adjust = "holm") ⑤
```

```
?corr.test()
```

- ① Choose one of your variables to be your x variable
- ② Choose the other to be your y variable
- ③ You can choose “pairwise” or “complete”. For information on what each does, use the following function to access the documentation: `?psych::corr.test()`
- ④ You can adjust method to be other ones like “spearman”
- ⑤ You can also use “bonferroni” among a few others

Below we will see the output of the correlation results as you might be used to seeing in a program like SPSS.

```
print(corr_results)
```

```
Call:corr.test(x = data$mpg, y = data$wt, use = "pairwise", method = "pearson",
              adjust = "holm")
```

```
Correlation matrix
```

```
[1] -0.87
```

```
Sample Size
```

```
[1] 32
```

```
These are the unadjusted probability values.
```

```
The probability values adjusted for multiple tests are in the p.adj object.
```

```
[1] 0
```

To see confidence intervals of the correlations, print with the `short=FALSE` option

While the above is great, notice we didn’t get a confidence interval output despite asking for it with `ci = TRUE`. Sometimes R will store complex computations within the output object (e.g., `corr_results`). To get this output we can put a `$` after the output. If there is extra information stored, but not shown, we’ll get a drop down box. We want the `ci` option. Below we will see the output that results from this. We should see the following:

$r = -.87$  ,  $p < .001$  with a CI = [-.93,-.74]

```
print(corr_results$ci)
```

	lower	r	upper	p
NA-NA	-0.9338264	-0.8676594	-0.7440872	1.293959e-10