431 Class 06

https://thomaselove.github.io/431-2024/

2024-09-12

## Today’s Agenda

* Assumptions of our uncertainty intervals
  + for one sample / paired differences
  + for two independent samples
* Transforming an Outcome (see [Chapter 7 of our book](https://thomaselove.github.io/431-book/))
  + Why transform?
  + The importance of the logarithm and other power transformations
  + Tukey’s ladder and the Box-Cox plot
  + Back-transforming predictions, not coefficients

## Load packages and set theme

library(janitor)  
  
library(car) ## new today  
library(infer) ## new today  
library(MKinfer)  
  
library(patchwork)  
library(rstanarm)  
library(easystats)  
library(tidyverse)  
  
theme\_set(theme\_bw())  
knitr::opts\_chunk$set(comment = NA)  
  
source("c06/data/Love-431.R") # for the lovedist() function

## Returning to the DM-464 data

dm6 <- read\_csv("c06/data/dm464\_class06.csv", show\_col\_types = FALSE) |>  
 janitor::clean\_names() |>  
 mutate(across(where(is.character), as\_factor)) |>  
 mutate(statin\_f = as\_factor(statin),  
 statin\_f = fct\_recode(statin\_f,   
 "Statin" = "1", "No Statin" = "0")) |>  
 mutate(id\_code = as.character(id\_code))  
  
dim(dm6)

[1] 464 9

## Uncertainty Interval Assumptions

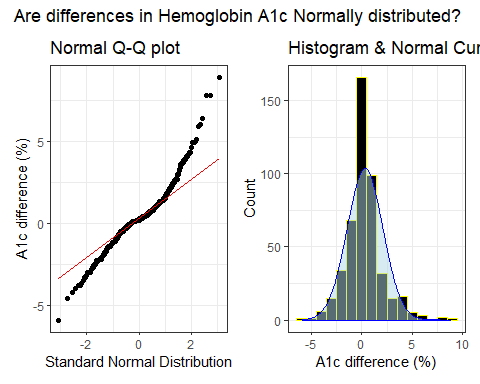
### One Sample (perhaps of Paired Differences)

* All approaches assume our sample is a **random** sample (or at least a representative one) from the population of interest.
* Ordinary Least Squares model / Paired T test also assumes that the population of interest follows a Normal distribution, so that our sample data should look as though it were drawn from a Normal distribution.
* Our Bayesian approach assumes a (weakly informative) prior distribution on the coefficient in our model.
* The bootstrap approach does not assume a Normal distribution for the population.

Main check in data: Normal distribution?

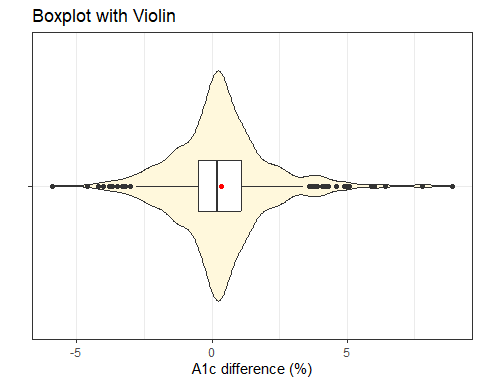
## Difference in A1c (end - baseline)

dm6 <- dm6 |> mutate(a1c\_diff = a1c\_end - a1c\_base)  
  
p1 <- ggplot(dm6, aes(sample = a1c\_diff)) +  
 geom\_qq() + geom\_qq\_line(col = "red") +  
 labs(y = "A1c difference (%)", x = "Standard Normal Distribution",  
 title = "Normal Q-Q plot")  
  
bw = 1 # specify width of bins in histogram  
  
p2 <- ggplot(dm6, aes(x = a1c\_diff)) +  
 geom\_histogram(binwidth = bw, fill = "black", col = "yellow") +  
 stat\_function(fun = function(x)   
 dnorm(x, mean = mean(dm6$a1c\_diff, na.rm = TRUE),   
 sd = sd(dm6$a1c\_diff, na.rm = TRUE)) \*   
 length(dm6$a1c\_diff) \* bw,  
 geom = "area", alpha = 0.5,   
 fill = "lightblue", col = "blue") +  
 labs(x = "A1c difference (%)", y = "Count",  
 title = "Histogram & Normal Curve")  
  
p1 + p2 +   
 plot\_annotation("Are differences in Hemoglobin A1c Normally distributed?")



## Boxplot for Difference in A1c

ggplot(dm6, aes(x = a1c\_diff, y = "")) +  
 geom\_violin(fill = "cornsilk") +  
 geom\_boxplot(width = 0.2) +  
 stat\_summary(fun = mean, geom = "point", shape = 16, col = "red") +  
 labs(y = "", x = "A1c difference (%)", title = "Boxplot with Violin")



## Does a Normal model fit well?

Do we have …

1. A histogram that is symmetric and bell-shaped.
2. A boxplot where the box is symmetric around the median, as are the whiskers, without severe outliers.
3. A normal Q-Q plot that essentially falls on a straight line.
4. If in doubt, maybe compare mean and sd to median and MAD, and consider Empirical Rule to help make tough calls.

* **Don’t** rely on hypothesis tests of whether data follow a Normal distribution.

## What to do about outliers?

The paired differences here appear to be symmetric, but with outliers.

* This should push us towards a method which doesn’t require the assumption of Normality.
  + Specifically, the bootstrap would likely be a better choice than a t test, although it didn’t make a meaningful difference when we ran it last time.
  + 90% uncertainty interval with OLS/t test was (0.20, 0.47) and the bootstrap 90% uncertainty interval was also (0.20, 0.47).
* Could we transform these data in a non-linear way to better match assumptions?
  + Not really, no. Most of the transformations we use regularly are designed to address skew in the data, rather than tail behavior.
* Should we build our interval around the median instead of the mean?
  + sample median = 0.2, sample mean = 0.33
  + Maybe, but does that actually address the issue in this case?

## Bootstrap 90% interval for Median

This is from the infer package…

set.seed(431)  
  
x\_med <- dm6 |>  
 observe(response = a1c\_diff, stat = "median")  
  
res1 <- dm6 |>  
 specify(response = a1c\_diff) |>  
 generate(reps = 2000, type = "bootstrap") |>  
 calculate(stat = "median") |>  
 get\_confidence\_interval(level = 0.90, type = "percentile")  
  
res1 <- res1 |>  
 mutate(pt\_est = x\_med$stat) |>  
 relocate(pt\_est)  
  
res1

# A tibble: 1 × 3  
 pt\_est lower\_ci upper\_ci  
 <dbl> <dbl> <dbl>  
1 0.200 0.200 0.300

## Uncertainty Interval Assumptions

### Two Independent Samples

* All methods assume each of the two samples is a random sample (or at least a representative one) from its population of interest.
* Ordinary Least Squares model / Pooled T test additionally assumes that:
  + **each** of the two populations of interest follows a Normal distribution, **and**
  + **either** the variance of those two populations is equal, so it makes sense to create a pooled estimate of the standard deviation, **or** the sample sizes are equal in the two groups (a balanced design)
* The Welch t test assumes Normality, but not equal variances.
* Our Bayesian approach assumes a (weakly informative) prior distribution on each of the coefficients (intercept and slope) of our model.
* The bootstrap approach does not assume a Normal distribution for the population, but can take advantage of an assumption of equal variances if it exists.

## From last class…

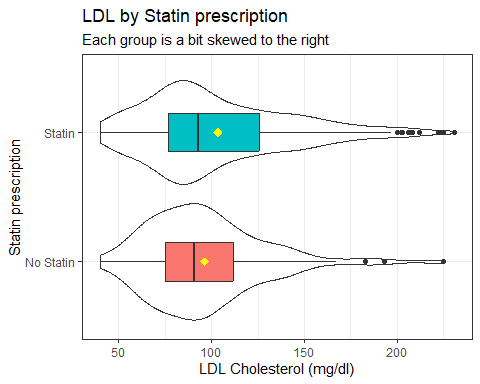
Estimating the mean difference in LDL levels at baseline for the “Statin” group minus the “No Statin” group…

| Approach | Estimate & 90% Interval |
| --- | --- |
| Ordinary least squares / Pooled t | 7.17 (0.02, 14.32) |
| Bayesian fit with stan\_glm() | 7.22 (-0.06, 14.56) |
| Welch t without pooling sd | 7.17 (0.86, 13.48) |
| Bootstrap with pooled sd | 7.21 (0.34, 14.29) |
| Bootstrap with unpooled sd | 7.23 (0.88, 13.66) |

* Is there a big impact here of using the Bootstrap rather than OLS?

## LDL by Statin prescription status

ggplot(dm6, aes(x = ldl\_base, y = statin\_f)) +  
 geom\_violin() +  
 geom\_boxplot(aes(fill = statin\_f), width = 0.3) +  
 stat\_summary(fun = mean, geom = "point",   
 shape = 18, size = 3, col = "yellow") +  
 guides(fill = "none") +  
 labs(y = "Statin prescription", x = "LDL Cholesterol (mg/dl)",  
 title = "LDL by Statin prescription",  
 subtitle = "Each group is a bit skewed to the right")



## Can we do something about the skew?

* Could we build a confidence interval for medians instead of means?
  + Yes, with the bootstrap, for example.
* Could we transform the data in a non-linear way to better match assumptions?
  + Yes, with the help of Tukey’s ladder of power transformations.

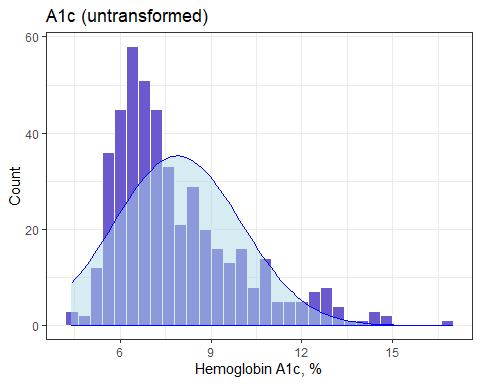
## Tukey’s Ladder

| Power () | -2 | -1 | -0.5 | 0 | 0.5 | 1 | 2 | 3 |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Transformation |  |  |  | log |  |  |  |  |

* Works to address skew, mainly, rather than symmetry with problems in the tails
* Some of these transformations require all values to be positive. If they aren’t, we can add a small amount to each observation until they are.
* Start at no transformation ()
* Try moving in one direction or the other, searching for a result which better matches what we might expect from a Normal distribution
  + If a step (like taking the square root) helps, move in that same direction.
  + If it doesn’t help, try moving in the other direction.

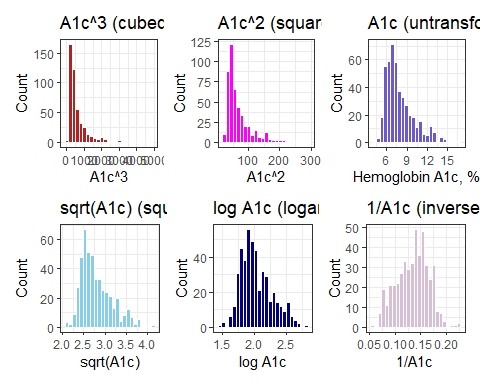
## A1c values (end of study)

ggplot(dm6, aes(x = a1c\_end)) +   
 geom\_histogram(binwidth = 0.4, fill = "slateblue", col = "white") +   
 stat\_function(fun = function(x)   
 dnorm(x, mean = mean(dm6$a1c\_end, na.rm = TRUE),   
 sd = sd(dm6$a1c\_end, na.rm = TRUE)) \*   
 length(dm6$a1c\_end) \* 0.4,  
 geom = "area", alpha = 0.5,   
 fill = "lightblue", col = "blue") +  
 labs(title = "A1c (untransformed)", x = "Hemoglobin A1c, %", y = "Count")



## Six Power Transformations

p1 <- ggplot(dm6, aes(x = a1c\_end^3)) +   
 geom\_histogram(bins = 25, fill = "firebrick", col = "white") +   
 labs(title = "A1c^3 (cubed)", x = "A1c^3", y = "Count")  
  
p2 <- ggplot(dm6, aes(x = a1c\_end^2)) +   
 geom\_histogram(bins = 25, fill = "magenta", col = "white") +   
 labs(title = "A1c^2 (squared)", x = "A1c^2", y = "Count")  
  
p3 <- ggplot(dm6, aes(x = a1c\_end)) +   
 geom\_histogram(bins = 25, fill = "slateblue", col = "white") +   
 labs(title = "A1c (untransformed)", x = "Hemoglobin A1c, %", y = "Count")  
  
p4 <- ggplot(dm6, aes(x = sqrt(a1c\_end))) +   
 geom\_histogram(bins = 25, fill = "skyblue", col = "white") +   
 labs(title = "sqrt(A1c) (square root)", x = "sqrt(A1c)", y = "Count")  
  
p5 <- ggplot(dm6, aes(x = log(a1c\_end))) +   
 geom\_histogram(bins = 25, fill = "navy", col = "white") +   
 labs(title = "log A1c (logarithm)", x = "log A1c", y = "Count")  
  
p6 <- ggplot(dm6, aes(x = 1/a1c\_end)) +   
 geom\_histogram(bins = 25, fill = "thistle", col = "white") +   
 labs(title = "1/A1c (inverse)", x = "1/A1c", y = "Count")  
  
(p1 + p2 + p3) / (p4 + p5 + p6)

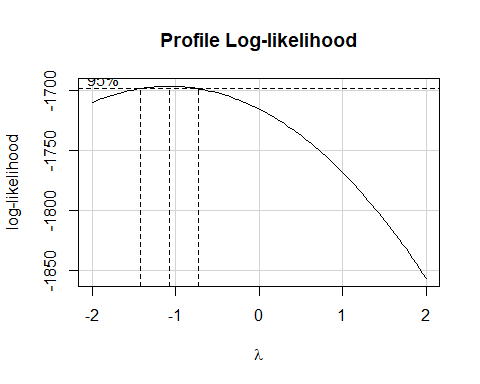


## Which Transformation to Choose?

Box-Cox approach: can we get a suggested “power” to use when transforming our outcome?

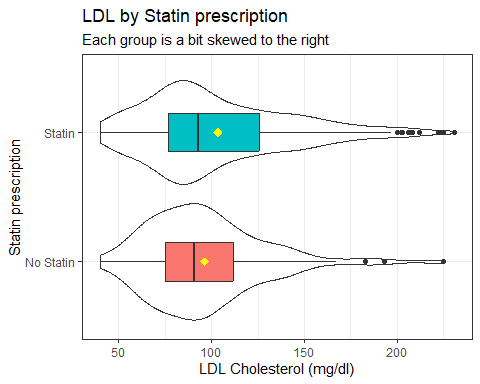
* Specify the model then apply boxCox from car package.
* Here, all values of our outcome (A1c) are strictly positive.
  + If not, we’d have to add a constant so that they were.

fit1 <- lm(a1c\_end ~ 1, data = dm6)  
boxCox(fit1)



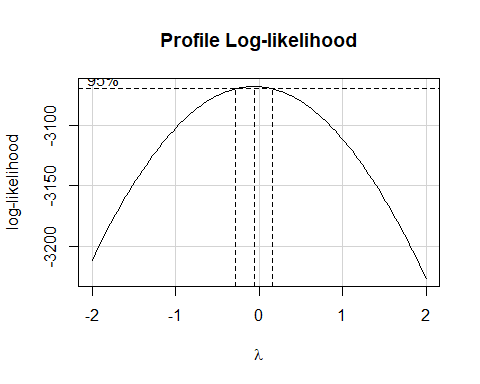
## Back to LDL by Statin prescription

ggplot(dm6, aes(x = ldl\_base, y = statin\_f)) +  
 geom\_violin() +  
 geom\_boxplot(aes(fill = statin\_f), width = 0.3) +  
 stat\_summary(fun = mean, geom = "point",   
 shape = 18, size = 3, col = "yellow") +  
 guides(fill = "none") +  
 labs(y = "Statin prescription", x = "LDL Cholesterol (mg/dl)",  
 title = "LDL by Statin prescription",  
 subtitle = "Each group is a bit skewed to the right")



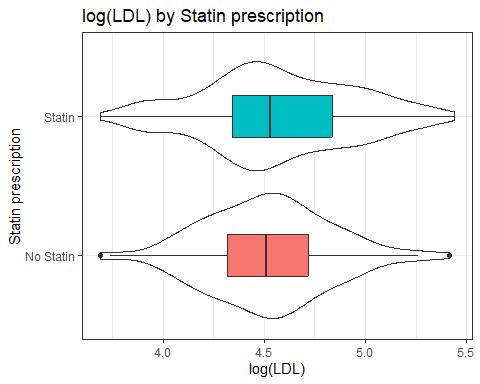
## What transformation should we try?

fit2 <- lm(ldl\_base ~ statin, data = dm6)  
boxCox(fit2)



## log(LDL) by Statin?

ggplot(dm6, aes(x = log(ldl\_base), y = statin\_f)) +  
 geom\_violin() + geom\_boxplot(aes(fill = statin\_f), width = 0.3) +  
 labs(y = "Statin prescription", x = "log(LDL)",  
 title = "log(LDL) by Statin prescription") +  
 guides(fill = "none")



## Estimate log(LDL) by Statin

Let’s use a 90% uncertainty interval…

fit3 <- lm(log(ldl\_base) ~ statin\_f, data = dm6)  
  
model\_parameters(fit3, ci = 0.90)

Parameter | Coefficient | SE | 90% CI | t(462) | p  
------------------------------------------------------------------------  
(Intercept) | 4.52 | 0.04 | [ 4.46, 4.58] | 122.75 | < .001  
statin f [Statin] | 0.05 | 0.04 | [-0.02, 0.12] | 1.23 | 0.220

Uncertainty intervals (equal-tailed) and p-values (two-tailed) computed  
 using a Wald t-distribution approximation.

The model has a log-transformed response variable. Consider using  
 `exponentiate = TRUE` to interpret coefficients as ratios.

## Expectations from our model

* The coefficients of this model exist in the transformed world, but we can make predictions back on our original scale.

estimate\_expectation(fit3, data = "grid", ci = 0.90)

Model-based Expectation  
  
statin\_f | Predicted | SE | 90% CI  
-------------------------------------------  
No Statin | 4.52 | 0.04 | [4.46, 4.58]  
Statin | 4.57 | 0.02 | [4.54, 4.60]  
  
Variable predicted: ldl\_base  
Predictors modulated: statin\_f

* and we exponentiate to get back to our original scale…

## Exponentiation of Expectations

* We back out of the logged predictions by exponentiating them.

| Group | Predicted LDL | 90% uncertainty interval |
| --- | --- | --- |
| No Statin | exp(4.52) = 91.8 | (exp(4.46), exp(4.58)) = (86.5, 97.5) |
| Statin | exp(4.57) = 96.5 | (exp(4.54), exp(4.60)) = (93.7, 99.5) |

* These intervals describe uncertainty about the **average predicted LDL across all subjects** in each statin group.

## Fitting Individual Predictions

* We use estimate\_prediction() to make predictions for an individual subject in each group.

estimate\_prediction(fit3, data = "grid", ci = 0.90)

Model-based Prediction  
  
statin\_f | Predicted | SE | 90% CI  
-------------------------------------------  
No Statin | 4.52 | 0.36 | [3.92, 5.12]  
Statin | 4.57 | 0.36 | [3.97, 5.17]  
  
Variable predicted: ldl\_base  
Predictors modulated: statin\_f

* and again we exponentiate to get back to our original scale…

## Fitting Individual Predictions

* We back out of the logged predictions by exponentiating them.

| Group | Predicted LDL | 90% uncertainty interval |
| --- | --- | --- |
| No Statin | exp(4.52) = 91.8 | (exp(3.92), exp(5.12)) = (50.4, 167.3) |
| Statin | exp(4.57) = 96.5 | (exp(3.97), exp(5.17)) = (53, 175.9) |

* These intervals describe uncertainty about an **individual predicted LDL for a single subject** within each statin group.

## Using the log transformation

* The log transformation in R (log in R is the natural log - the base 10 log is log10()) generates coefficients which describe a multiplicative effect.

If we fit a model to predict using of the form:

where is the natural logarithm, then if we exponentiate the slope coefficient, we get the multiplicative factor for each one-unit increase in the predictor.

## The Math

$$
\log(y) = \beta\_0 + \beta\_1 x \\
\exp(\log(y)) = \exp(\beta\_0 + \beta\_1 x) \\
y = \exp(\beta\_0 + \beta\_1 x) \\
y = \exp(\beta\_0) \times \exp(\beta\_1 x)
$$

This implies that our predictor () has a multiplicative relationship with our outcome () instead of the usual additive relationship. Hence, we can express the effect of a one-unit change in on as a percentage change.

## Example 1

Suppose we build a regression model with equation .

Note that .

* If Harry’s value of is 1 point larger than Sally’s, the predicted value for Harry will be increased by 26% relative to Sally’s. (Increased by 26% = multiplied by 1.26.)

## Example 2

fit3 <- lm(log(ldl\_base) ~ statin\_f, data = dm6)  
  
model\_parameters(fit3, ci = 0.90)

Parameter | Coefficient | SE | 90% CI | t(462) | p  
------------------------------------------------------------------------  
(Intercept) | 4.52 | 0.04 | [ 4.46, 4.58] | 122.75 | < .001  
statin f [Statin] | 0.05 | 0.04 | [-0.02, 0.12] | 1.23 | 0.220

Uncertainty intervals (equal-tailed) and p-values (two-tailed) computed  
 using a Wald t-distribution approximation.

* We can exponentiate these coefficients…

model\_parameters(fit3, exponentiate = TRUE, ci = 0.90)

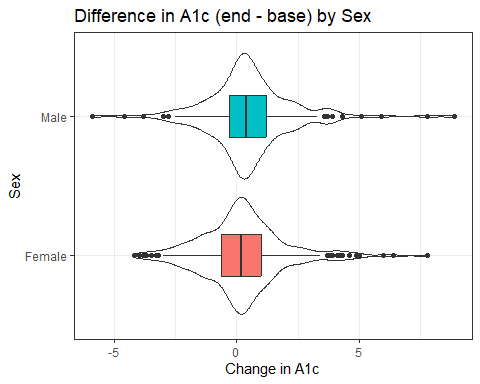
Parameter | Coefficient | SE | 90% CI | t(462) | p  
-------------------------------------------------------------------------  
(Intercept) | 91.67 | 3.37 | [86.27, 97.40] | 122.75 | < .001  
statin f [Statin] | 1.05 | 0.04 | [ 0.98, 1.13] | 1.23 | 0.220

Uncertainty intervals (equal-tailed) and p-values (two-tailed) computed  
 using a Wald t-distribution approximation.

Subjects *with* a statin prescription have 5% higher LDL, on average, than do subjects *without*. The ratio has point estimate 1.05 and 90% interval estimate (0.98, 1.13).

## Difference in A1c by Sex

ggplot(dm6, aes(x = a1c\_diff, y = sex)) +  
 geom\_violin() + geom\_boxplot(aes(fill = sex), width = 0.3) +  
 labs(x = "Change in A1c", y = "Sex",  
 title = "Difference in A1c (end - base) by Sex") +  
 guides(fill = "none")



## Numerical Summary

dm6 |> group\_by(sex) |>   
 reframe(lovedist(a1c\_diff)) |>   
 print\_md(digits = 3)

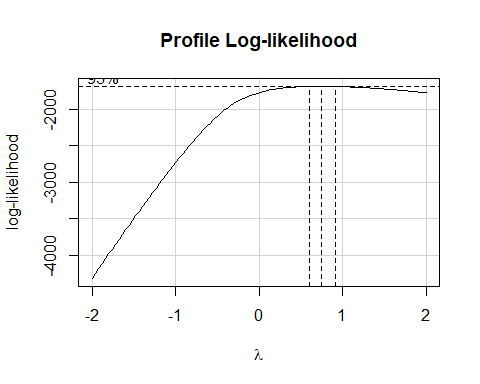
| sex | n | miss | mean | sd | med | mad | min | q25 | q75 | max |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Female | 288 | 0 | 0.227 | 1.747 | 0.200 | 1.186 | -4.200 | -0.625 | 1.000 | 7.800 |
| Male | 176 | 0 | 0.503 | 1.838 | 0.400 | 1.186 | -5.900 | -0.300 | 1.200 | 8.900 |

* Observed difference in sample means is 0.503 - 0.227 = 0.276
* Do we have a balanced design?
* Do we feel comfortable pooling these standard deviations?
* Do we feel comfortable assuming Normality for each sex?

## Might a transformation help here?

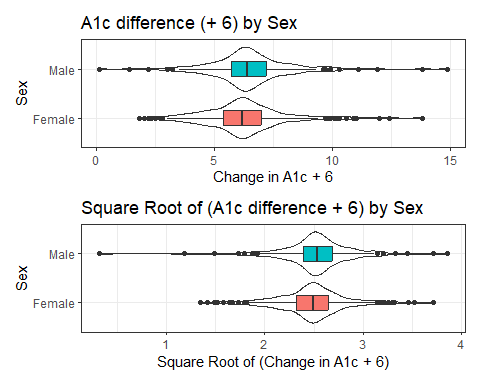
Some of the changes in A1c are negative (minimum was -5.9)

dm6 <- dm6 |> mutate(a1c\_diff\_p6 = a1c\_diff + 6)  
fit4 <- lm(a1c\_diff\_p6 ~ sex, data = dm6)  
boxCox(fit4)



## Does using a square root help us here?

p1 <- ggplot(dm6, aes(x = a1c\_diff\_p6, y = sex)) +  
 geom\_violin() + geom\_boxplot(aes(fill = sex), width = 0.3) +  
 labs(x = "Change in A1c + 6", y = "Sex",  
 title = "A1c difference (+ 6) by Sex") +  
 guides(fill = "none")  
  
p2 <- ggplot(dm6, aes(x = sqrt(a1c\_diff\_p6), y = sex)) +  
 geom\_violin() + geom\_boxplot(aes(fill = sex), width = 0.3) +  
 labs(x = "Square Root of (Change in A1c + 6)", y = "Sex",  
 title = "Square Root of (A1c difference + 6) by Sex") +  
 guides(fill = "none")  
  
p1 / p2



## Addressing outliers?

### OLS model

fit5 <- lm(a1c\_diff ~ sex, data = dm6)  
  
model\_parameters(fit5, ci = 0.90)

Parameter | Coefficient | SE | 90% CI | t(462) | p  
-----------------------------------------------------------------  
(Intercept) | 0.23 | 0.11 | [ 0.05, 0.40] | 2.16 | 0.031  
sex [Male] | 0.28 | 0.17 | [ 0.00, 0.56] | 1.62 | 0.105

Uncertainty intervals (equal-tailed) and p-values (two-tailed) computed  
 using a Wald t-distribution approximation.

* Male subjects had A1c changes that were 0.28 larger than females, on average, with 90% uncertainty interval (0, 0.56) according to our OLS model (pooled t test.)

## Addressing outliers?

### Bayesian model

set.seed(20240912)  
fit6 <- stan\_glm(a1c\_diff ~ sex, data = dm6, refresh = 0)  
  
model\_parameters(fit6, ci = 0.90)

Parameter | Median | 90% CI | pd | Rhat | ESS | Prior  
---------------------------------------------------------------------------------------  
(Intercept) | 0.22 | [ 0.05, 0.39] | 98.42% | 1.000 | 3666.00 | Normal (0.33 +- 4.46)  
sexMale | 0.28 | [ 0.00, 0.57] | 94.80% | 1.000 | 3649.00 | Normal (0.00 +- 9.19)

Uncertainty intervals (equal-tailed) and p-values (two-tailed) computed  
 using a MCMC distribution approximation.

* Male subjects had A1c changes that were 0.28 larger than females, on average, with 90% uncertainty interval (0, 0.57) according to our Bayesian model.

## Addressing outliers?

### Bootstrap with boot.t.test

set.seed(20240912)  
boot.t.test(a1c\_diff ~ sex, data = dm6, conf.level = 0.90)

Bootstrap Welch Two Sample t-test  
  
data: a1c\_diff by sex  
number of bootstrap samples: 9999  
bootstrap p-value = 0.1096   
bootstrap difference of means (SE) = -0.2787276 (0.1715815)   
90 percent bootstrap percentile confidence interval:  
 -0.566107955 0.002070707  
  
Results without bootstrap:  
t = -1.6028, df = 355.56, p-value = 0.1099  
alternative hypothesis: true difference in means is not equal to 0  
90 percent confidence interval:  
 -0.561349458 0.008003498  
sample estimates:  
mean in group Female mean in group Male   
 0.2267361 0.5034091

## Session Information

xfun::session\_info()

R version 4.4.1 (2024-06-14 ucrt)  
Platform: x86\_64-w64-mingw32/x64  
Running under: Windows 11 x64 (build 22631)  
  
Locale:  
 LC\_COLLATE=English\_United States.utf8   
 LC\_CTYPE=English\_United States.utf8   
 LC\_MONETARY=English\_United States.utf8  
 LC\_NUMERIC=C   
 LC\_TIME=English\_United States.utf8   
  
Package version:  
 abind\_1.4-5 arrangements\_1.1.9 askpass\_1.2.0   
 backports\_1.5.0 base64enc\_0.1-3 bayesplot\_1.11.1   
 bayestestR\_0.14.0 BH\_1.84.0.0 bit\_4.0.5   
 bit64\_4.0.5 blob\_1.2.4 boot\_1.3-31   
 broom\_1.0.6 bslib\_0.8.0 cachem\_1.1.0   
 callr\_3.7.6 car\_3.1-2 carData\_3.0-5   
 cellranger\_1.1.0 checkmate\_2.3.2 cli\_3.6.3   
 clipr\_0.8.0 coda\_0.19-4.1 codetools\_0.2-20   
 colorspace\_2.1-1 colourpicker\_1.3.0 commonmark\_1.9.1   
 compiler\_4.4.1 conflicted\_1.2.0 correlation\_0.8.5   
 cowplot\_1.1.3 cpp11\_0.5.0 crayon\_1.5.3   
 crosstalk\_1.2.1 curl\_5.2.2 data.table\_1.16.0   
 datasets\_4.4.1 datawizard\_0.12.3 DBI\_1.2.3   
 dbplyr\_2.5.0 Deriv\_4.1.3 desc\_1.4.3   
 digest\_0.6.37 distributional\_0.4.0 doBy\_4.6.22   
 dplyr\_1.1.4 DT\_0.33 dtplyr\_1.3.1   
 dygraphs\_1.1.1.6 easystats\_0.7.3 effectsize\_0.8.9   
 emmeans\_1.10.4 estimability\_1.5.1 evaluate\_0.24.0   
 exactRankTests\_0.8-35 fansi\_1.0.6 farver\_2.1.2   
 fastmap\_1.2.0 fontawesome\_0.5.2 forcats\_1.0.0   
 foreach\_1.5.2 fs\_1.6.4 gargle\_1.5.2   
 generics\_0.1.3 ggplot2\_3.5.1 ggridges\_0.5.6   
 glmnet\_4.1-8 glue\_1.7.0 gmp\_0.7-5   
 googledrive\_2.1.1 googlesheets4\_1.1.1 graphics\_4.4.1   
 grDevices\_4.4.1 grid\_4.4.1 gridExtra\_2.3   
 gtable\_0.3.5 gtools\_3.9.5 haven\_2.5.4   
 highr\_0.11 hms\_1.1.3 htmltools\_0.5.8.1   
 htmlwidgets\_1.6.4 httpuv\_1.6.15 httr\_1.4.7   
 ids\_1.0.1 igraph\_2.0.3 infer\_1.0.7   
 inline\_0.3.19 insight\_0.20.4 isoband\_0.2.7   
 iterators\_1.0.14 janitor\_2.2.0 jomo\_2.7-6   
 jquerylib\_0.1.4 jsonlite\_1.8.8 knitr\_1.48   
 labeling\_0.4.3 later\_1.3.2 lattice\_0.22-6   
 lazyeval\_0.2.2 lifecycle\_1.0.4 lme4\_1.1-35.5   
 loo\_2.8.0 lubridate\_1.9.3 magrittr\_2.0.3   
 markdown\_1.13 MASS\_7.3-61 Matrix\_1.7-0   
 MatrixModels\_0.5.3 matrixStats\_1.4.0 memoise\_2.0.1   
 methods\_4.4.1 mgcv\_1.9.1 mice\_3.16.0   
 miceadds\_3.17-44 microbenchmark\_1.5.0 mime\_0.12   
 miniUI\_0.1.1.1 minqa\_1.2.8 mitml\_0.4-5   
 mitools\_2.4 MKdescr\_0.8 MKinfer\_1.2   
 modelbased\_0.8.8 modelr\_0.1.11 multcomp\_1.4-26   
 munsell\_0.5.1 mvtnorm\_1.3-1 nlme\_3.1-164   
 nloptr\_2.1.1 nnet\_7.3-19 numDeriv\_2016.8.1.1   
 openssl\_2.2.1 ordinal\_2023.12.4.1 pan\_1.9   
 parallel\_4.4.1 parameters\_0.22.2 patchwork\_1.2.0   
 pbkrtest\_0.5.3 performance\_0.12.3 pillar\_1.9.0   
 pkgbuild\_1.4.4 pkgconfig\_2.0.3 plyr\_1.8.9   
 posterior\_1.6.0 prettyunits\_1.2.0 processx\_3.8.4   
 progress\_1.2.3 promises\_1.3.0 ps\_1.7.7   
 purrr\_1.0.2 quantreg\_5.98 QuickJSR\_1.3.1   
 R6\_2.5.1 ragg\_1.3.2 rappdirs\_0.3.3   
 RColorBrewer\_1.1.3 Rcpp\_1.0.13 RcppArmadillo\_14.0.0.1  
 RcppEigen\_0.3.4.0.2 RcppParallel\_5.1.9 readr\_2.1.5   
 readxl\_1.4.3 rematch\_2.0.0 rematch2\_2.1.2   
 report\_0.5.9 reprex\_2.1.1 reshape2\_1.4.4   
 rlang\_1.1.4 rmarkdown\_2.28 rpart\_4.1.23   
 rstan\_2.32.6 rstanarm\_2.32.1 rstantools\_2.4.0   
 rstudioapi\_0.16.0 rvest\_1.0.4 sandwich\_3.1-0   
 sass\_0.4.9 scales\_1.3.0 see\_0.9.0   
 selectr\_0.4.2 shape\_1.4.6.1 shiny\_1.9.1   
 shinyjs\_2.1.0 shinystan\_2.6.0 shinythemes\_1.2.0   
 snakecase\_0.11.1 sourcetools\_0.1.7.1 SparseM\_1.84.2   
 splines\_4.4.1 StanHeaders\_2.32.10 stats\_4.4.1   
 stats4\_4.4.1 stringi\_1.8.4 stringr\_1.5.1   
 survival\_3.7-0 sys\_3.4.2 systemfonts\_1.1.0   
 tensorA\_0.36.2.1 textshaping\_0.4.0 TH.data\_1.1-2   
 threejs\_0.3.3 tibble\_3.2.1 tidyr\_1.3.1   
 tidyselect\_1.2.1 tidyverse\_2.0.0 timechange\_0.3.0   
 tinytex\_0.52 tools\_4.4.1 tzdb\_0.4.0   
 ucminf\_1.2.2 utf8\_1.2.4 utils\_4.4.1   
 uuid\_1.2.1 V8\_5.0.0 vctrs\_0.6.5   
 viridisLite\_0.4.2 vroom\_1.6.5 withr\_3.0.1   
 xfun\_0.47 xml2\_1.3.6 xtable\_1.8-4   
 xts\_0.14.0 yaml\_2.3.10 zoo\_1.8-12