m8

Will Tirone

Load Data

```r
# some cleaning steps to get the data in shape to work with

school_dat = read.csv('school.csv') |>
  pivot_longer(
    cols = -c(school, num_classes),
    names_to = "class",
    values_to = "mean_score"
  ) |>
  filter(!is.na(mean_score)) |>
  mutate(class = str_replace(class, "_[1234]$", ""))
```

1)

a)

The outcome of interest is a given class' mean test score on a math exam, noting that they're standardized and scaled here. The study (as presented in the reading) focuses on teacher-level analysis in relation to math scores. Say we want to know whether or not the class size has an effect on the math scores. In school 1, regular class 1, we observed a score of -0.197. We want to know if this teacher would have the same outcome with a smaller class, which is impossible to observe since the teacher can only be assigned to one class. The Fisher sharp null states that the teacher would have the same outcome in a smaller class (across all schools and classes). Then, we randomly permute the treatments and outcomes, calculate our statistic, and approximate a p-value.

b)

Here, we calculate:

$$T^{dif,\lambda_{RSS}} = \sum_{i=1}^{16} \frac{\text{num classes in school i}}{\text{total classes}} (\bar{y}^{obs}_{small}(i) - \bar{y}^{obs}_{reg.}(i)) = 0.228, \qquad \text{for school i = 1,...,16}$$

And note that since the question asks for a one-tailed test, I drop the absolute value bars and just use the raw statistic.

```
total_classes = 68

# computes y_bar_treatment - y_bar_control
# so y_bar for small class - y_bar for regular class
y_bars = school_dat |>
    group_by(school, class) |>
    summarise(y_bar = mean(mean_score)) |>
    group_by(school) |>
    summarise(diff_y_bar = diff(y_bar))

# get number of classes
c = school_dat |>
    dplyr::select(school, num_classes) |>
    unique()

# combine and do calculations
left_join(y_bars, c, by = c("school")) |>
  mutate(T_stat_i = (num_classes/total_classes)*diff_y_bar) |>
  group_by() |>
  summarise(T_stat = sum(T_stat_i)) |>
  pull()
```

```
[1] 0.2278897
```

c)

The classes were previously assigned regular, regular, small, small. Below using `sample()` and a vector with the assignments, we permute them randomly and get small, regular, regular, small. We will substitute these values in to the existing data frame and compute the statistic again. Then reusing code from part b), we have an updated statistic of:

$$T^{dif,\lambda_{RSS}} = 0.263$$

```r
# randomly sample
set.seed(154)
class = c("reg_class", "reg_class", "small_class", "small_class")
new_assign = sample(class)

altered = school_dat |>
    filter(school == 16) |>
    mutate(class = new_assign)

school_new_assign = bind_rows(
  school_dat |> filter(school != 16),
  altered
)

# copying code we used previously to compute

# computes y_bar_treatment - y_bar_control
# so y_bar for small class - y_bar for regular class
y_bars = school_new_assign |>
    group_by(school, class) |>
    summarise(y_bar = mean(mean_score)) |>
    group_by(school) |>
    summarise(diff_y_bar = diff(y_bar))

# get number of classes
c = school_new_assign |>
    dplyr::select(school, num_classes) |>
    unique()

# combine and do calculations
left_join(y_bars, c, by = c("school")) |>
  mutate(T_stat_i = (num_classes/total_classes)*diff_y_bar) |>
  group_by() |>
  summarise(T_stat = sum(T_stat_i)) |>
  pull()
```

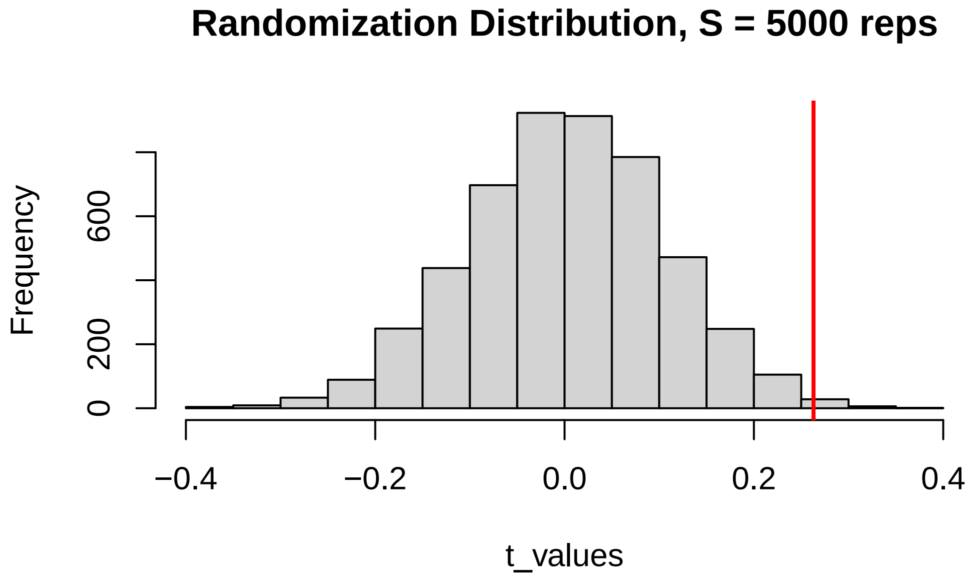
```
[1] 0.2628897
```

d)

Here, I'm using a similar approach to previous questions, but repeat the process 5,000 times. Within each school, we randomize the small vs. regular, and compute and store the $T^{diff,\lambda_{RSS}}$ for each iteration. Then, we check plot the distribution and compute an empirical p-value of ≈ 0.006 depending on the random seed. Since this is well below 0.05, we can safely reject the Fisher sharp null that the treatment (reg. vs. small classes) has no effect on mean math test scores.

```r
t_values = c()
S = 5000

for (i in 1:S){

  subset = data.frame()

  for (j in 1:16){
    school_j = school_dat[school_dat$school == j, ]
    school_j$class = sample(school_j$class)
    subset = bind_rows(subset, school_j)
  }

  # copying code we used previously to compute

  # computes y_bar_treatment - y_bar_control
  # so y_bar for small class - y_bar for regular class
  y_bars = subset |>
      group_by(school, class) |>
      summarise(y_bar = mean(mean_score)) |>
      group_by(school) |>
      summarise(diff_y_bar = diff(y_bar))

  # get number of classes
  c = subset |>
      dplyr::select(school, num_classes) |>
      unique()

  # combine and do calculations
  T_stat = left_join(y_bars, c, by = c("school")) |>
      mutate(T_stat_i = (num_classes/total_classes)*diff_y_bar) |>
      group_by() |>
      summarise(T_stat = sum(T_stat_i)) |>
```

```
        pull()

  t_values = c(t_values, T_stat)
}

hist(t_values,
     main='Randomization Distribution, S = 5000 reps')
abline(v = 0.263, col = 'red', lwd=2)
```



And computing the p-value:

```
mean(t_values >= 0.263)
```

```
[1] 0.0062
```

2)

First, we'll do model selection by comparing the additive model with the model with an interaction between school:class. The p-value for the F-test is not significant, so we use the simpler model, `mean_score ~ class + school`, where class is the treatment here (the values are `small_class` and `reg_class`).

```r
m1 = lm(mean_score ~ class + as.factor(school), data = school_dat)
m2 = lm(mean_score ~ class * as.factor(school), data = school_dat)
anova(m1, m2)
```

```
Analysis of Variance Table

Model 1: mean_score ~ class + as.factor(school)
Model 2: mean_score ~ class * as.factor(school)
  Res.Df    RSS Df Sum of Sq      F Pr(>F)
1     51 8.3547
2     36 5.2204 15    3.1343 1.4409 0.1812
```

Below I report the point estimates, standard errors, and 95% confidence intervals for our final model. In addition, R^2 and the regression variance are:

$$R^2 = 0.6292$$
$$\sigma^2 = 0.1638$$

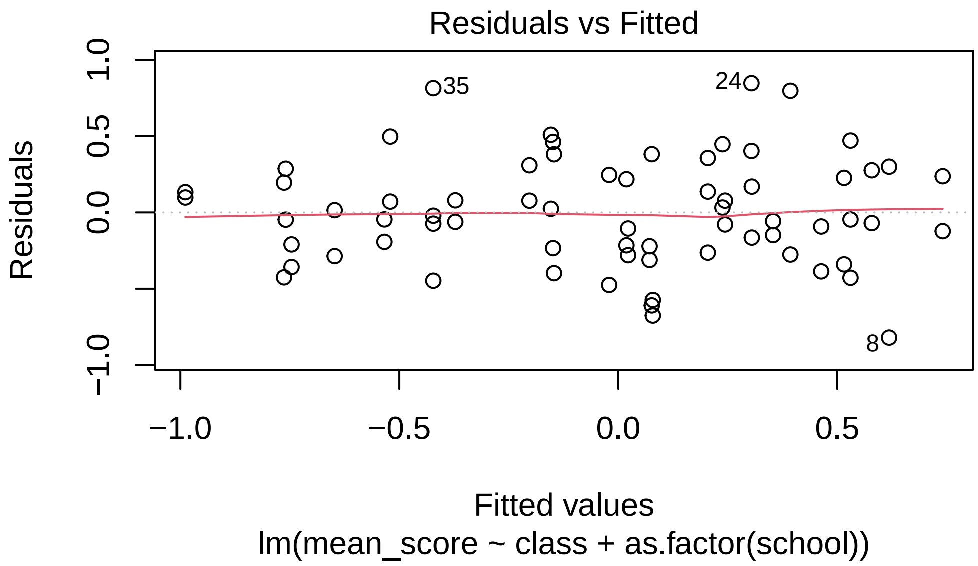
```r
coefs = cbind(Estimate = coef(m1),
              SE = sqrt(diag(vcov(m1))),
              suppressMessages(confint(m1)))

coefs |> kable()
```

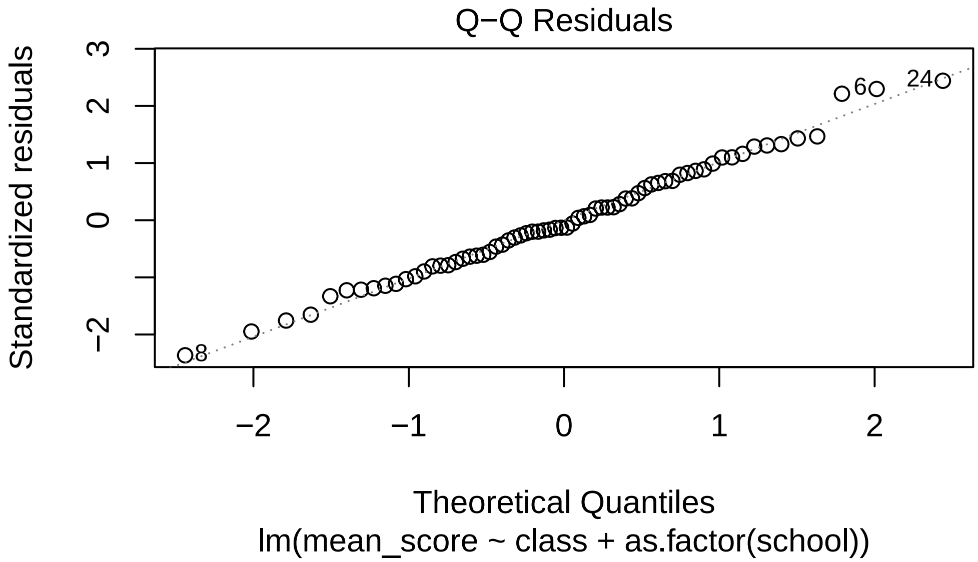
	Estimate	SE	2.5 %	97.5 %
(Intercept)	0.0185797	0.2083306	-0.3996615	0.4368208
classsmall_class	0.2253406	0.0989436	0.0267030	0.4239783
as.factor(school)2	0.3745000	0.2861964	-0.2000632	0.9490632
as.factor(school)3	-0.0393841	0.2716899	-0.5848244	0.5060563
as.factor(school)4	-0.7647500	0.2861964	-1.3393132	-0.1901868
as.factor(school)5	-0.2215000	0.2861964	-0.7960632	0.3530632
as.factor(school)6	0.0602500	0.2861964	-0.5143132	0.6348132
as.factor(school)7	0.2195000	0.2861964	-0.3550632	0.7940632
as.factor(school)8	-0.6663068	0.2617803	-1.1918526	-0.1407609
as.factor(school)9	-1.0072500	0.2861964	-1.5818132	-0.4326868
as.factor(school)10	-0.7780000	0.2861964	-1.3525632	-0.2034368
as.factor(school)11	-0.1675000	0.2861964	-0.7420632	0.4070632
as.factor(school)12	0.2864159	0.2716899	-0.2590244	0.8318563
as.factor(school)13	0.3350000	0.2861964	-0.2395632	0.9095632
as.factor(school)14	0.4970000	0.2861964	-0.0775632	1.0715632
as.factor(school)15	-0.3907500	0.2861964	-0.9653132	0.1838132
as.factor(school)16	-0.1725000	0.2861964	-0.7470632	0.4020632

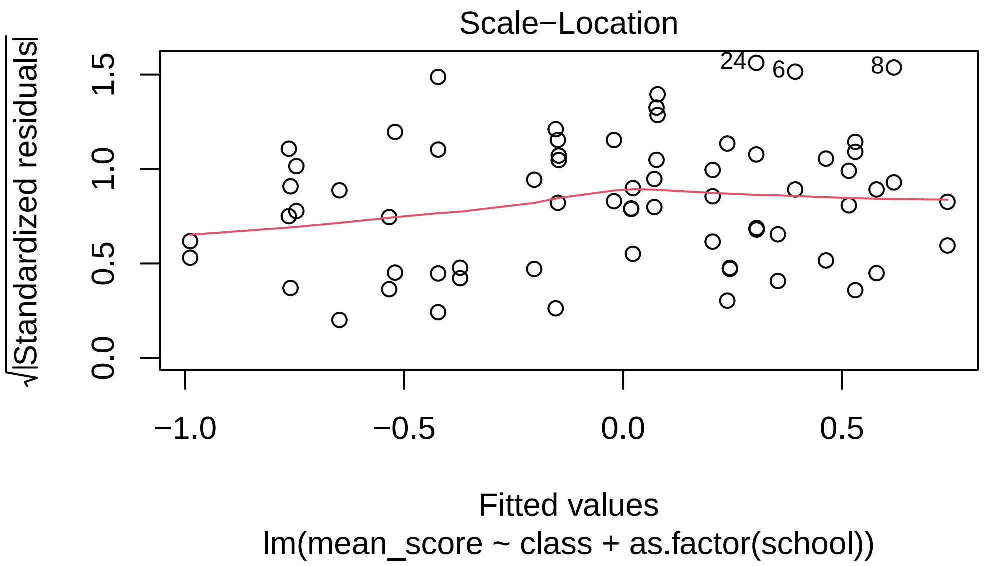
Checking model assumptions:

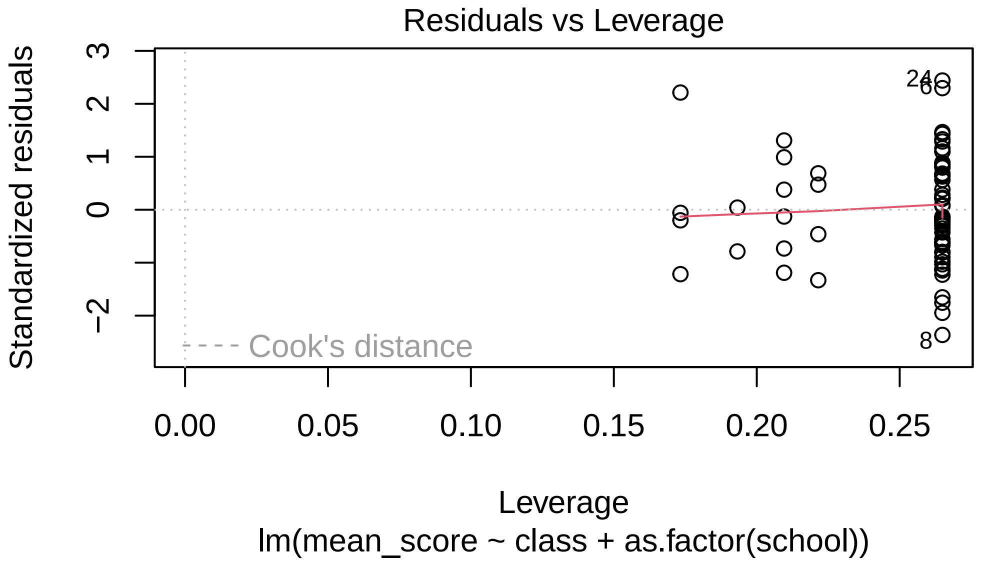
Since we only have a handful of points for each school, I think it makes sense to look at all the data points together. First, it looks like there's no evidence of heteroskedasticity, the residuals don't have an obvious pattern over the fitted values. Second, the residuals look normally distributed based on the QQ plot. We don't have any reason to believe that the linearity or independence assumptions are broken either, so we'll conclude that the modeling assumptions aren't broken.

```
plot(m1)
```


Residuals vs Fitted
Residuals
1.0
0.5
0.0
−1.0
35
24
8
−1.0
−0.5
0.0
0.5
Fitted values
lm(mean_score ~ class + as.factor(school))


Q−Q Residuals
Standardized residuals
3
2
1
0
−2
6
24
8
−2
−1
0
1
2
Theoretical Quantiles
lm(mean_score ~ class + as.factor(school))





3)

a)

Our colleague is incorrect because they're not considering `Block == 0`, which has a coefficient of 0. So this is significantly different than the values for blocks 1 and 2.

b)

This is true. It seems like the treatment has no effect since a 95% conf. interval would cover zero for both interaction terms.

c)

Blocking is not done randomly, it is done intentionally based on background characteristics that we want to balance. It is nonsensical to consider being "randomly" assigned to a block, so we don't agree with our colleague.

4)

a)

Computed the coefficients here by plugging in values and computing.

$$\beta_0 = 1$$
$$\beta_1 = 4$$
$$\beta_2 = 2$$
$$\beta_3 = 0$$

b)

Since the interaction coefficient is zero, we conclude there is no interaction among A and B. Now, we can conclude that we do have enough information to estimate the effects from part a) because the blocking + randomization covers every possible case. That is, we have participants in `A=0, B=0`, `A=1, B=0`, and `A=0, B=1`. We don't care about having a group with `A=1, B=1` because this interaction has no effect.