Problem Set 2

Alex, Micah, Scott, and David

12/01/2021

Contents

1 What happens when pilgrims attend the Hajj pilgrimage to Mecca?

1.1 State a null hypothesis

State the sharp-null hypothesis that you will be testing.

```
# Sharp null hypothesis: Attending Hajj has no impact on views toward others.
```

Answer: Sharp null hypothesis: Attending Hajj has no impact on views toward others.

1.2 Group by average

Using `data.table`, group the data by `success` and report whether views toward others are generally more positive among lottery winners or lottery non-winners. This answer should be of the form `d[ , .(mean_views = ...), keyby = ...]` where you have filled in the `...` with the appropriate functions and variables.

```
# the result should be a data.table with two columns and two rows
hajj_group_mean <- d[, .(mean_views = mean(views, na.rm = TRUE)), keyby = success]

# from the `hajj_group_mean` produce a single, numeric vector that is the ate.
# check that it is numeric using `class(hajj_ate)`
hajj_ate <- hajj_group_mean[success == 1, mean_views] - hajj_group_mean[success == 0, mean_views]

hajj_group_mean
```

```
## Key: <success>
##    success mean_views
##      <int>      <num>
## 1:       0   1.868304
## 2:       1   2.343137
```

```
hajj_ate
```

```
## [1] 0.4748337
```

Answer: Views toward others are generally more positive among lottery winners compared to non-winners, as the mean views score is higher for those who attended Hajj (2.343137) than those who did not (1.868304).

```
## do your work to conduct the randomization inference here.
## as a reminder, RI will randomly permute / assign the treatment variable
## and recompute the test-statistic (i.e. the mean difference) under each permutation
## this should be a numeric vector that has a length equal to the number
## of RI permutations you ran
n_permutations <- 10000
hajj_ri_distribution <- numeric(n_permutations)

for (i in 1:n_permutations) {
  d[, permuted_success := sample(success)]
  permuted_group_mean <- d[, .(mean_views = mean(views, na.rm = TRUE)), keyby = permuted_success]
  hajj_ri_distribution[i] <- permuted_group_mean[permuted_success == 1, mean_views] - permuted_group_me
}
```

1.3 Randomization inference: At least as large

C. How many of the simulated random assignments generate an estimated ATE that is at least as large as the actual estimate of the ATE? Conduct your work in the code chunk below, saving the results into hajj_count_larger, but also support your coding with a narrative description. In that narrative description (and throughout), use R's "inline code chunks" to write your answer consistent with each time your run your code.

```r
# length 1 numeric vector from comparison of `hajj_ate` and `hajj_ri_distribution`
hajj_count_larger <- sum(hajj_ri_distribution >= hajj_ate)
hajj_count_larger
```

```
## [1] 20
```

Answer: 20

1.4 Randomization inference: one-sided p-value

If there are hajj_count_larger (20) randomizations that are larger than hajj_ate (0.4748337), what is the *one-tailed* p-value? Both write the code in the following chunk, and include a narrative description of the result following your code.

```r
# length 1 numeric vector
hajj_one_tailed_p_value <- hajj_count_larger / n_permutations
hajj_one_tailed_p_value
```

```
## [1] 0.002
```

Answer: 0.002

1.5 Randomization inference: two-sided p-value

Now, conduct a similar test, but for a two-sided p-value. You can either use two tests, one for larger than and another for smaller than; or, you can use an absolute value (abs). Both write the code in the following chunk, and include a narrative description of the result following your code.

```r
# length 1 numeric vector
hajj_two_tailed_p_value <- sum(abs(hajj_ri_distribution) >= abs(hajj_ate)) / n_permutations
hajj_two_tailed_p_value
```

```
## [1] 0.0045
```

Answer: 0.0042

2 Sports Cards

2.1 t-test and confidence interval

Using a `t.test`, compute a 95% confidence interval for the difference between the treatment mean and the control mean. After you conduct your test, write a narrative statement, *using inline code evaluation* that describes what your tests find, and how you interpret these results. (You should be able to look into `str(t_test_cards)` to find the pieces that you want to pull to include in your written results.)

```
# this should be the t.test object. Extract pieces from this object
# in-text below the code chunk.
setnames(d, "uniform_price_auction", "treatment")

t_test_cards <- t.test(bid ~ treatment, data = d)
t_test_cards
```

```
##
##  Welch Two Sample t-test
##
## data:  bid by treatment
## t = 2.8211, df = 61.983, p-value = 0.006421
## alternative hypothesis: true difference in means between group 0 and group 1 is not equal to 0
## 95 percent confidence interval:
##   3.557141 20.854624
## sample estimates:
## mean in group 0 mean in group 1
##        28.82353        16.61765
```

Narrative Analysis: ...

2.2 Interpretation of confidence interval

In your own words, what does this confidence interval mean? This can be simple language, but it has to be statistically appropriate language.

Answer: The t-test results indicate that the mean difference in bids between the treatment and control auction formats is $28.82353 - 16.61765 = 12.20588$. The confidence interval suggests that the true difference in means is between 3.5571 and 20.8546, meaning that we are 95% confident that the true difference in bids between the treatment and control groups falls within this range. Since the confidence interval does not include zero and the p-value (0.0064) is less than 0.05, this provides strong evidence that the treatment auction format led to significantly lower bids than the control auction format.

2.3 Randomization inference, and confidence interval?

Conduct a randomization inference process, with `n_ri_loops = 1000`, using an estimator that you write by hand (i.e. in the same way as earlier questions). On the sharp-null distribution that this process creates, compute the 2.5% quantile and the 97.5% quantile using the function `quantile` with the appropriate vector passed to the `probs` argument. This is the randomization-based uncertainty that is generated by your design. After you conduct your test, write a narrative statement of your test results.

```
## first, do you work for the randomization inference
n_ri_loops <- 1000

cards_ate             <- mean(d[treatment == 1, bid]) - mean(d[treatment == 0, bid])
cards_ri_distribution <- numeric(n_ri_loops) # numeric vector of length equal

for (i in 1:n_ri_loops) {
```

```
  d[, permuted_treatment := sample(treatment)]
  permuted_group_mean <- d[, .(mean_bid = mean(bid, na.rm = TRUE)), keyby = permuted_treatment]
  cards_ri_distribution[i] <- permuted_group_mean[permuted_treatment == 1, mean_bid] - permuted_group_m
} # to your number of RI permutations

cards_ri_quantiles    <- quantile(cards_ri_distribution, probs = c(0.025, 0.975)) # there's a built-in
cards_ri_p_value      <- mean(abs(cards_ri_distribution) >= abs(cards_ate))

cards_ri_p_value
```

```
## [1] 0.005
```

```
cards_ri_quantiles[1]
```

```
##      2.5%
## -9.147059
```

```
cards_ri_quantiles[2]
```

```
##    97.5%
## 9.558824
```

Narrative: ... The randomization inference approach provides a confidence interval of -9.030882 to 9.208824, which is based on resampling under the null hypothesis. The p-value is 0.005, which suggests that the observed difference in bids is statistically significant if the p-value is below 0.05.

2.4 Compare regression and randomization inference

Do you learn anything different if you regress the outcome on a binary treatment variable? To answer this question, regress `bid` on a binary variable equal to 0 for the control auction and 1 for the treatment auction and then calculate the 95% confidence interval using *classical standard errors* (in a moment you will calculate with *robust standard errors*). There are two ways to do this – you can code them by hand; or use a built-in, `confint`. After you conduct your test, write a narrative statement of your test results.

```
# this should be a model object, class = 'lm'.
mod <- lm(bid ~ treatment, data = d)
summary(mod)  # Display regression results
```

```
##
## Call:
## lm(formula = bid ~ treatment, data = d)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -28.824 -11.618  -3.221   8.382  58.382
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   28.824      3.059   9.421 7.81e-14 ***
## treatment    -12.206      4.327  -2.821  0.00631 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 17.84 on 66 degrees of freedom
## Multiple R-squared:  0.1076, Adjusted R-squared:  0.09409
## F-statistic: 7.959 on 1 and 66 DF,  p-value: 0.006315
```

```r
confint(mod)  # Display confidence intervals
```

```
##                 2.5 %    97.5 %
## (Intercept)  22.71534 34.931716
## treatment   -20.84416 -3.567603
```

Narrative: ... The ordinary least squares (OLS) regression results indicate that the estimated treatment effect is -12.206, meaning that the treatment auction format leads to lower bids compared to the control format. The 95% confidence interval for this effect is (-20.844, -3.568), suggesting that the true effect is likely negative, supporting the theoretical prediction.

Comparing this to the randomization inference results, we see that the confidence interval from randomization inference is (-9.030882, 9.208824), which is wider and includes zero, unlike the regression confidence interval. This suggests that randomization inference provides a more conservative estimate of uncertainty and does not rule out the possibility that the treatment effect could be negligible.

Additionally, the p-value from the OLS regression (0.00631) is quite similar to the randomization inference p-value (0.005), indicating strong evidence against the null hypothesis in both approaches.

2.5 Regression with robust confidence interval

Calculate the 95% confidence interval using robust standard errors, using the `sandwich` package. There is a function in `lmtest` called `coefci` that can help with this. It is also possible to do this work by hand. After you conduct your test, write a narrative statement of your test results.

```r
# this should be a numeric vector of length 2
cards_robust_ci <- coefci(mod, vcov. = vcovHC(mod, type = "HC1"))
cards_robust_ci
```

```
##                 2.5 %    97.5 %
## (Intercept)  21.98187 35.665186
## treatment   -20.84416 -3.567603
```

Narrative: ... Using robust standard errors, the confidence interval for the treatment effect changes to -20.84416 to -3.567603, accounting for heteroskedasticity. This suggests that our findings are robust to assumptions about variance homogeneity.

2.6 Compare and contrast results

Characterize what you learn from each of these different methods – are the results contingent on the method of analysis that you choose?

Answer: Both methods show a statistically significant treatment effect, with similar p-values confirming the result is unlikely due to chance. Differences in confidence intervals highlight the impact of method choice.

3 Power Analysis

3.1 Describe your testing procedure

Describe a t-test based testing procedure that you might conduct for this experiment. What is your null hypothesis, and what would it take for you to reject this null hypothesis? (This second statement could either be in terms of p-values, or critical values.)

Answer: The t-test procedure will compare the means of the treatment and control groups. The null hypothesis states that there is no difference in means between the two groups. We reject the null hypothesis if the p-value from the t-test is below 0.05, indicating a statistically significant difference.

3.2 Suppose you only had ten subjects, what would you learn

Suppose that you are only able to recruit 10 people to be a part of your experiment – 5 in treatment and another 5 in control. Simulate "re-conducting" the sports card experiment once by sampling from the data you previously collected, and conducting the test that you've written down in part 1 above. *Given the results of this 10 person simulation, would your test reject the null hypothesis?*

```
d <- fread('../data/list_data_2019.csv')
setnames(d, "uniform_price_auction", "treatment")

t_test_ten_people <- t.test(bid ~ treatment, data = d[sample(.N, 10, replace = TRUE)])
t_test_ten_people
```

```
##
##  Welch Two Sample t-test
##
## data:  bid by treatment
## t = 1.9786, df = 5.0966, p-value = 0.1037
## alternative hypothesis: true difference in means between group 0 and group 1 is not equal to 0
## 95 percent confidence interval:
##  -7.645431 60.045431
## sample estimates:
## mean in group 0 mean in group 1
##            38.2            12.0
```

Answer: the p-value is 0.3716, so we fail to reject the null hypothesis.

3.3 With only ten subjects, what is your power?

Repeat this process – sampling 10 people from your existing data and conducting the appropriate test – one-thousand times. Each time that you conduct this sample and test, pull the p-value from your t-test and store it in an object for later use. *Consider whether your sampling process should sample with or without replacement.*

```
# fill this in with the p-values from your power analysis
t_test_p_values <- rep(NA, 1000)

for (i in 1:1000) {
  valid_sample <- FALSE

  while (!valid_sample) {
    sample_data <- d[sample(.N, 10, replace = TRUE)]

    # Count observations in each group
    group_counts <- table(sample_data$treatment)
```

```r
    # Ensure both groups exist and have at least 2 observations
    if (length(group_counts) == 2 && all(group_counts >= 2)) {
      valid_sample <- TRUE
    }
  }

  # Perform t-test after verifying valid sample
  t_test <- t.test(bid ~ treatment, data = sample_data)
  t_test_p_values[i] <- t_test$p.value
}

t_test_p_values
```

```
##     [1] 0.0246085597 0.7492025936 0.4812380984 0.1198824348 0.0850709694
##     [6] 0.8262169362 0.3154079157 0.0877564445 0.1966264197 0.1892137743
##    [11] 0.2312692273 0.3003610514 0.2084645301 0.5063622731 0.2277792399
##    [16] 0.1940245903 0.0372577232 0.5668412216 0.5327714536 0.4942255963
##    [21] 0.5748956440 0.9542815125 0.5459775685 0.4230168814 0.4963513265
##    [26] 0.4676325883 0.1707998547 0.8099317967 0.1702753621 0.1533957446
##    [31] 0.2610664322 0.9679786398 0.0307102738 0.3186697649 0.4486924379
##    [36] 0.1020552339 0.0222465712 0.3259492094 0.9182537753 0.9594375072
##    [41] 0.9368053056 0.9333982388 0.1270815132 0.6675433912 0.7634507301
##    [46] 0.0173360955 0.0387020811 0.3661902174 0.3868762989 0.5078705006
##    [51] 0.8649372923 0.2593705786 0.8225473281 0.9317025886 0.0503577613
##    [56] 0.0797808393 0.8786942662 0.0365592617 0.4769863744 0.4396514829
##    [61] 0.0012979362 0.9888179015 0.7444603430 0.6075461128 0.3499883030
##    [66] 0.0527805112 0.4877623341 0.0270556978 0.3319875146 0.3871530460
##    [71] 0.8122385979 0.0443139891 0.2122498928 0.4202938249 0.0332626407
##    [76] 0.4931654187 0.3860666744 0.0248583024 0.2415889426 0.5709653510
##    [81] 0.6489065883 0.0865094894 0.2613737700 0.4698547161 0.0139927978
##    [86] 0.3395560372 0.0245146947 0.5181349514 0.0557086228 0.6991676005
##    [91] 0.7804650212 0.5152377777 0.7755370329 0.3321846227 0.7205012310
##    [96] 0.2661324471 0.1560549523 0.0105778618 0.1400363407 0.8675833996
##   [101] 0.0169938853 0.1867923853 0.3480818845 0.5023392184 0.0041990028
##   [106] 0.1904418825 0.8940700812 0.9445940424 0.0288051678 0.0039475785
##   [111] 0.0864685859 0.0183099504 0.4862891694 0.6241808078 0.8198817151
##   [116] 0.0018606069 0.2422024242 0.1864024743 0.4311309519 0.5136919147
##   [121] 0.1845210633 0.1355698897 0.7660622554 0.5099883220 0.6510498693
##   [126] 0.7387593244 0.3078953879 0.1976176975 0.4324683418 0.8029109361
##   [131] 0.1122647918 0.3108976620 0.8869396081 0.0280844643 0.0742593127
##   [136] 0.0190986969 0.2623898203 0.5294139163 0.9115712597 0.2700282164
##   [141] 0.1326118098 0.1718452903 0.0010061836 0.7977794156 0.6117264154
##   [146] 0.1474741632 0.6902659678 0.6961717339 0.6146770556 0.0152549426
##   [151] 0.1094659856 0.0333231007 0.0199082147 0.2552284048 0.1147759769
##   [156] 0.0517203999 0.3757522034 0.0799645082 0.0271788444 0.0662158717
##   [161] 0.7382536759 0.6563453954 0.0334573734 0.9181832942 0.1916126784
##   [166] 0.7389522230 0.3530225736 0.8112631074 0.1661265723 0.8776245581
##   [171] 0.7180497178 0.1274269277 0.3712989179 0.0477747710 0.1037903927
##   [176] 0.3390267102 0.1525057289 0.4052663206 0.1854662806 0.6811321213
##   [181] 0.3485550819 0.3036138388 0.1843731991 0.7071224820 0.0009936004
##   [186] 0.0823930159 0.5628027122 0.0391025979 0.8657878312 0.3712854988
##   [191] 0.4871363632 0.0117612535 0.0249414630 0.0224528239 0.9555619468
##   [196] 0.0784424480 0.0938122595 0.1913601156 0.1542506094 0.8994961681
```

```
## [201] 0.4922091269 0.4459112105 0.1535168868 0.1266317288 0.4588955716
## [206] 0.1553491212 0.0120283382 0.4194473535 0.4294767224 0.2486243195
## [211] 0.0541584470 0.1392097443 0.1587713317 0.0198494407 0.4403544611
## [216] 0.0795671152 0.2087277625 0.7020153798 0.2231532892 0.3490716063
## [221] 0.1380865657 0.5341841581 0.3814765774 0.1239122066 0.6415487573
## [226] 0.6987034629 0.6502595064 0.6501526137 0.1605484077 0.8477543903
## [231] 0.0465205494 0.0501118384 0.6145618317 0.0127658925 0.6473920247
## [236] 0.0831525993 0.0012875414 0.9463595742 0.4829980260 0.0529891179
## [241] 0.1930531743 0.0329247556 1.0000000000 0.3199617151 0.4005397892
## [246] 0.1215660364 0.3524536315 0.4148733270 0.0984896649 0.9700822376
## [251] 0.6958443389 0.6624412551 0.2523222937 0.2222043695 0.5498574907
## [256] 0.1874840384 0.2356671160 0.6410259341 0.0499960367 0.1300040243
## [261] 0.0020357065 0.6874016855 0.0236236961 0.0452376390 0.2906678758
## [266] 0.3225558530 0.8466840032 0.0428845846 0.0810468419 0.7806883941
## [271] 0.0269387864 0.3405131522 0.8509411229 0.2177456568 0.5878345195
## [276] 0.0545436704 0.1443727902 0.0428892338 0.8617424230 0.0120666513
## [281] 0.2576488239 0.8976529348 0.0085447291 0.0722109781 0.0970966318
## [286] 0.5573100261 0.0989117620 0.5055281921 0.8189987307 0.3404119797
## [291] 0.0826036205 0.3248735218 0.3876892515 0.3462647832 0.6125151442
## [296] 0.6284056456 0.8987612022 0.0435628807 0.6190362667 0.2244532233
## [301] 0.5578106891 0.0659503106 0.7485734415 0.0738457145 0.3589008868
## [306] 0.6736179472 0.3389053782 0.4978757339 0.0527040498 0.0716144840
## [311] 0.4210415020 0.0587483765 0.0674906971 0.0279580982 0.9749671575
## [316] 0.0380071974 0.5099938567 0.4826052584 0.5355862275 0.4245711095
## [321] 0.3002803496 0.0198107308 0.0176130821 0.0464721559 0.0245725645
## [326] 0.1076186135 0.1957412180 0.1463360880 0.7487410900 0.5775265256
## [331] 0.1420659730 0.1943261915 0.0125845222 0.3883902431 0.4336493897
## [336] 0.9292465501 0.0284457771 0.2910575084 0.2569538215 0.2421055232
## [341] 0.0305720152 0.4941506851 0.9191316158 0.9151607932 0.1394602713
## [346] 0.9622545485 0.8962476892 0.2241703425 0.2239045335 0.6342791526
## [351] 0.7600262470 0.9775245610 0.9113422189 0.2654279452 0.0969714588
## [356] 0.2760815308 0.8944670840 0.0981261932 0.4403406939 0.7512137386
## [361] 0.1134463668 0.1127107591 0.1354162025 0.8086230072 0.6667870891
## [366] 0.9196353710 0.1216035964 0.3974238770 0.9589110354 0.6401884029
## [371] 0.4723715850 0.6641164115 0.2188501241 0.7578623398 0.1449136673
## [376] 0.8223752987 0.0357324208 0.0016657035 0.7348043276 0.0954652615
## [381] 0.6163767312 0.9227100247 0.1091690068 0.6711505580 0.0396824992
## [386] 0.7953134775 0.1146345120 0.1738186969 0.1111491378 0.2060006428
## [391] 0.1273954076 0.1842761150 0.0137666561 0.9239446838 0.2746795723
## [396] 0.6073341063 0.1710520788 0.3799940889 0.2029882419 0.0228871398
## [401] 0.3929456581 0.2643748623 0.0386675515 0.5115320054 0.0387749677
## [406] 0.0814203290 0.3581844000 0.4295418577 0.6187312726 0.1416168664
## [411] 0.2883133096 0.8191194292 0.1931198478 0.0304164827 0.3596838274
## [416] 0.1363943616 0.2223530278 0.3920991398 0.1911427148 0.1303492181
## [421] 0.0307288850 0.8935553924 0.1082238768 0.2830075759 0.5028057122
## [426] 0.2171330528 0.1830574370 0.1064863164 0.4438400934 0.5695424311
## [431] 0.0638565199 0.0929240263 0.7962773062 0.8677700962 0.0470826150
## [436] 0.0331204679 0.1179725113 0.4022110551 0.0386409186 0.6161272702
## [441] 0.2814843342 0.0384186964 0.2722928142 0.1698226988 0.4614647583
## [446] 0.3610214541 0.2513279598 0.9067585748 0.1196525089 0.4134811927
## [451] 0.8248195911 0.4090747839 0.2060239782 0.2289345073 0.3690190827
## [456] 0.5887798552 0.2350870609 0.7962219322 0.1965047206 0.1765681150
## [461] 0.9261145797 0.5103704424 0.4366488113 0.1348827067 0.0763495709
## [466] 0.0469287115 0.0642189601 0.8670728746 0.3645062779 0.1334890512
```

```
## [471] 0.1406108774 0.6050353173 0.0839014033 0.1420945120 0.4027453601
## [476] 0.7285848207 0.3862802275 0.6483211400 0.0033457618 0.2597061334
## [481] 0.0047183149 0.1439935777 0.2836593208 0.9023280082 0.3297710689
## [486] 0.3859251669 0.7201178203 0.2934646391 0.1186093657 0.0045434138
## [491] 0.1551852254 0.9376554478 0.2497232114 0.8854455929 0.1680490975
## [496] 0.0290102876 0.7683188026 0.2109156001 0.1039457964 0.3236940951
## [501] 0.4017710602 0.8607148696 0.8566473971 0.8586924551 0.0807289080
## [506] 1.0000000000 0.1461044396 0.3647935304 0.7342787331 0.5247009769
## [511] 0.0356698979 0.7041926283 0.0685390354 0.4657941057 0.1348095847
## [516] 0.0689761220 0.9763518240 0.6120038074 0.3547878379 0.7167243769
## [521] 0.1847165295 0.4781682212 0.2968370771 0.0322276000 0.1440334019
## [526] 0.6151331913 0.0076185873 0.1936914243 0.0790615910 0.8587825588
## [531] 0.1178837808 0.3029533220 0.0301297065 0.4336170443 0.0751724492
## [536] 0.6440342356 0.3223978929 0.3456145746 0.0216099678 0.3096139543
## [541] 0.2055511955 0.9036949809 0.9683898569 0.8500165217 0.4485118053
## [546] 0.9073708155 0.0146429173 0.0642889298 0.4043593637 0.0636561162
## [551] 0.1862947869 0.7800876313 0.3398778829 0.0853804521 0.1207191023
## [556] 0.5440340361 0.0929755257 1.0000000000 0.0410192547 0.4285242453
## [561] 0.2634775929 0.2374652688 0.7395303607 0.7502949937 0.1728118561
## [566] 0.0342398065 0.1605334347 0.6674607298 0.4788095357 0.9006253159
## [571] 0.7665790214 0.4995959532 0.0180692842 0.0051856439 0.2404500354
## [576] 0.3679664294 0.9129976955 0.2136815479 0.0345707660 0.0201470182
## [581] 0.2133831564 0.7985255405 0.9654896045 0.2447883236 0.2793064696
## [586] 0.9297215165 0.5179118592 0.2588170235 0.0128715694 0.3360644352
## [591] 0.0052561441 0.4127508877 0.9935075896 0.9206352565 0.0461243970
## [596] 0.2359740357 0.7688169632 0.6489434251 0.2154846170 0.7837379307
## [601] 0.8329549403 0.0684715670 0.0379985084 0.2375201123 0.1983221449
## [606] 0.0162726737 0.1216087920 0.1033541442 0.0046342821 0.4697787288
## [611] 0.1761559658 0.4273190727 0.0803979839 0.0175266565 0.3632875973
## [616] 0.0876822376 0.0972783654 0.0811017386 0.5971195642 0.7353363051
## [621] 0.0928986576 0.6491203784 0.0090064101 0.0784398395 0.1931686520
## [626] 0.5152688922 0.6810758387 0.4688747280 0.3263879309 0.6495840522
## [631] 0.3564402260 0.3906174543 0.1543584300 0.2242669427 0.5195535155
## [636] 0.0552749571 0.5141478378 0.2877442839 0.0547838140 0.5195644542
## [641] 0.9502989974 0.6685244202 0.6465268430 0.2008864973 0.8936409513
## [646] 0.0071617124 0.0870568188 0.3754318329 0.2931541733 0.8231896966
## [651] 0.0781089014 0.2059899028 0.7557783468 0.4089619777 0.2257738944
## [656] 0.2513894500 0.2277866522 0.7907451748 0.6673208909 0.4520830274
## [661] 0.0327801064 0.9607857206 0.0404205052 0.0726721506 0.6147202084
## [666] 0.1650776620 0.1093343242 0.6687370192 0.8906331298 0.5113068943
## [671] 0.0397173411 0.8417181376 0.1034922464 0.7533909760 0.2886441206
## [676] 0.3524512648 0.8693371160 0.1152440112 0.1011191903 0.8683161475
## [681] 0.9488423692 0.5788378927 0.1492340552 0.7491268390 0.0181609328
## [686] 0.7360707167 0.0998970683 0.2425187808 0.1330762116 0.0713358647
## [691] 0.3781140260 0.6406141410 0.0927419606 0.7215885207 0.4083544776
## [696] 0.4747682135 0.6982408813 0.4790061246 0.7850885289 0.5333860950
## [701] 0.0771947908 0.0383456980 0.7814329063 0.9616627253 0.0290310892
## [706] 0.0630707593 0.1345196044 0.1842851240 0.5169968498 0.9716224127
## [711] 0.0335216053 0.1180138470 0.5386722328 0.0846565235 0.0926680975
## [716] 0.8232436856 0.5340530120 0.8708734603 0.0406744478 0.1662934406
## [721] 0.6054789407 0.0175328527 0.8544278930 0.4171108913 0.7249175290
## [726] 0.2268065016 0.1757770199 0.0115738922 0.7267519269 0.4355581142
## [731] 0.0063236193 0.4378065361 0.3800808178 0.5764211750 0.1375308528
## [736] 0.8294409206 0.4333193118 0.1195119804 0.4632411106 0.0138153667
```

```
## [741] 0.0382883918 0.8562971533 0.1548174191 0.6747453089 0.0203133657
## [746] 0.0721850243 0.0155709598 0.2265094036 0.8685683275 0.2524224057
## [751] 0.8592681876 0.6612791051 0.1642708577 0.6060614126 0.0280655332
## [756] 0.3795632222 0.0663511310 0.1969864940 0.1752741531 0.4132377694
## [761] 0.6621553234 0.6668510684 0.4166465410 0.5513492799 0.2808996638
## [766] 0.2040458391 0.1928536100 0.3835920164 0.9905475039 0.0605503843
## [771] 0.3308198960 0.3370758581 0.4918112534 0.0038898252 0.8970594786
## [776] 0.9229226050 0.0799122256 0.1125690033 0.2682300697 0.3990218984
## [781] 0.1661634172 0.0021507520 0.8885467057 0.9662771771 0.3676088339
## [786] 0.2261556852 0.9319020289 0.0076992284 0.0279881416 0.0809056962
## [791] 0.0529556105 0.0300483383 0.6464740791 0.0210348329 0.5816784443
## [796] 0.9231180775 0.1861707817 0.4561945712 0.6120509083 0.2569935925
## [801] 0.2063863361 0.4918119199 0.7251851550 0.7684765092 0.6935865827
## [806] 0.5199918146 0.0340739385 0.0856349961 0.0288871792 0.3243252266
## [811] 0.0227393219 0.2690097682 0.3155114544 0.0764589405 0.0502482040
## [816] 0.0362573962 0.2997610343 0.2322761499 0.0990141054 0.0768776279
## [821] 0.0095002954 0.5710633998 0.7560016088 0.4838168600 0.0359759185
## [826] 0.4494448378 0.1389604759 0.0250165431 0.8928174232 0.0392363364
## [831] 0.5772151564 0.0027941237 0.9333303923 0.1300657592 0.1763704135
## [836] 0.2252028735 0.2647328868 0.6593029138 0.2678925452 0.1922464722
## [841] 0.0647382920 0.6949280242 0.3500448226 0.5281422159 0.0120843986
## [846] 0.2306969034 0.1904595367 0.6623907602 0.1279772033 0.2831852134
## [851] 0.8352331793 0.1412904016 0.7776231681 0.0045986125 0.0775790405
## [856] 0.0355796773 0.2195340933 0.3072818321 0.0328651510 0.3912418758
## [861] 0.0074398819 0.0065269360 0.8892809768 0.5897483656 0.0048402538
## [866] 0.3316516532 0.0319636938 0.0340659948 0.2224177945 0.3787078789
## [871] 0.6052880679 0.0013787217 0.0011804786 0.5347087998 0.5198062726
## [876] 0.7142901524 0.0472601851 0.0500077630 0.2963458909 0.9191585993
## [881] 0.1137207389 0.7535304566 0.0180498618 0.0671308290 0.1422069181
## [886] 0.1915384475 0.4876099288 0.2952110335 0.0219372993 0.0681895100
## [891] 0.2383490165 0.0623485596 0.7964923026 0.0902732096 0.5855320072
## [896] 0.9163596227 0.3015065580 0.0094114606 0.2535536083 0.5235659147
## [901] 0.4463146235 0.3001804484 0.1891712508 0.5431393533 0.3572023944
## [906] 0.6381610098 0.0158488108 0.5456398789 0.9012953345 0.6299369191
## [911] 0.1183590147 0.7391801856 0.4906736390 0.2082981820 0.0617006334
## [916] 0.0430323938 0.7028293082 0.0044986443 0.2641043452 0.7862479565
## [921] 0.0099188214 0.3367093783 0.7234267518 0.6934204733 0.7606455941
## [926] 0.2401890361 0.3761958259 0.0298568511 0.4174658443 0.0864677568
## [931] 0.8814906072 0.1716655319 0.7667494026 0.7524055783 0.5144801635
## [936] 0.2778702197 0.0642187238 0.1471006697 0.1321955454 0.0526989651
## [941] 0.4264290853 0.7612742346 0.2345663671 0.0110248786 0.8993983081
## [946] 0.1816407143 0.1007030435 0.1360659595 0.0511657237 0.2448459543
## [951] 0.0064117353 0.5947462356 0.9849451361 0.0374681791 0.7155645231
## [956] 0.3775868518 0.1353861550 0.2193178570 0.2027641614 0.7858174298
## [961] 0.8541732216 0.8645159019 0.4512307896 0.3007065948 0.9311846410
## [966] 0.0254067957 0.0792460021 0.2779652852 0.1693343194 0.4221950356
## [971] 0.4529195616 0.2177209923 0.0031332403 0.0279104216 0.0133352753
## [976] 0.0319421831 0.2566562574 0.3501500443 0.8613748466 0.0637824092
## [981] 0.0774854844 0.5822873674 0.5655390537 0.4316252245 0.7585482817
## [986] 0.7168020675 0.5691930831 0.2075190572 0.0508762586 0.7448985897
## [991] 0.5840009953 0.0529887315 0.1290840278 0.3659388401 0.6336564907
## [996] 0.9755273594 0.2781450026 0.7525460388 0.3332268275 0.0800270194
```

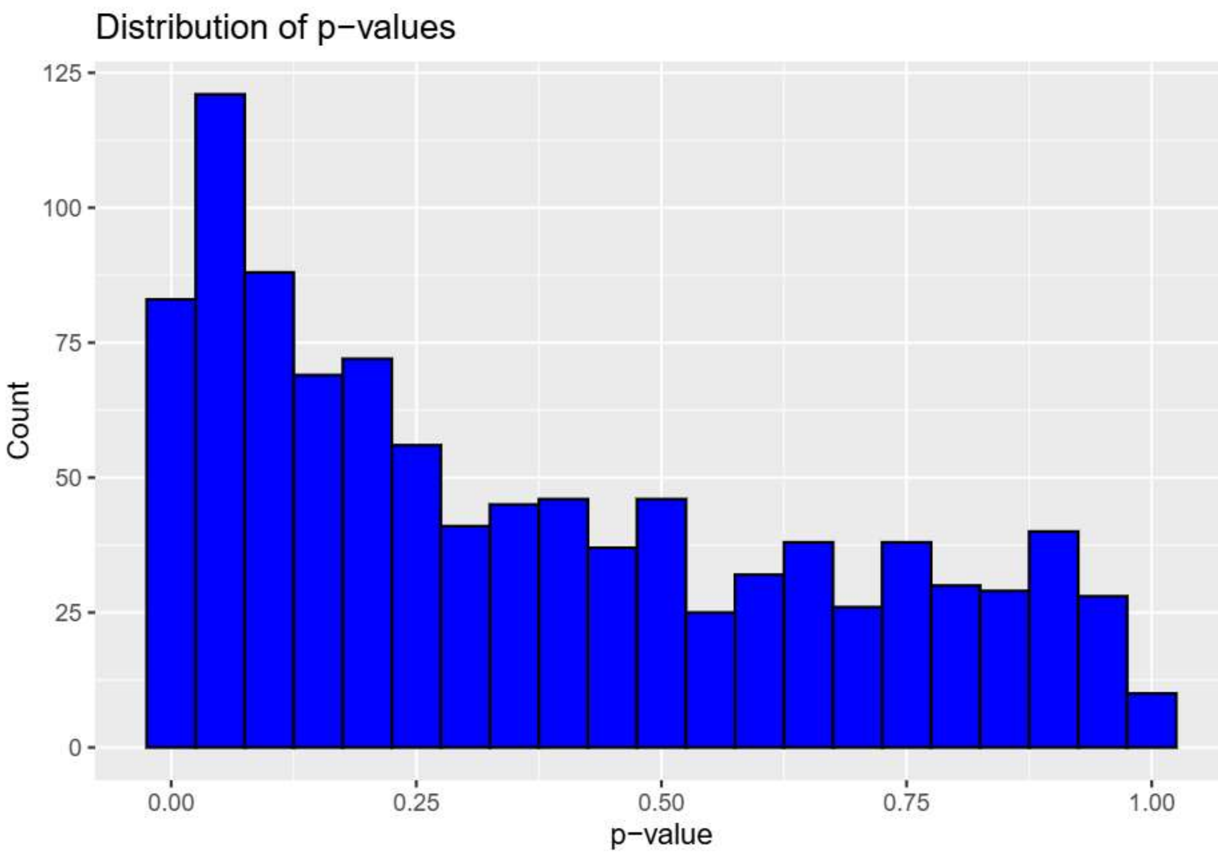
```
## you can either write a for loop, use an apply method, or use replicate
## (which is an easy-of-use wrapper to an apply method)
```

Answer: The vector t_test_p_values now contains the p-values from 1,000 simulations of the experiment with 10 subjects. These p-values will be used to calculate the power of the test.

3.4 Visual analysis

Use `ggplot` and either `geom_hist()` or `geom_density()` to produce a distribution of your p-values, and describe what you see. *What impression does this leave you with about the power of your test?*

```
library(ggplot2)
ggplot(data.frame(p_value = t_test_p_values), aes(x = p_value)) +
  geom_histogram(binwidth = 0.05, fill = "blue", color = "black") +
  labs(title = "Distribution of p-values", x = "p-value", y = "Count")
```



Answer: This distribution suggests that the test has low power with the current sample size (10 subjects, 5 per group). The test is not consistently able to detect a true effect, as indicated by the substantial number of p-values greater than 0.05. This leaves the impression that the design, with such a small sample size, is insufficient for reliable hypothesis testing.

3.5 Interpret your results, given your power

Suppose that you and David were to actually run this experiment and design – sample 10 people, conduct a t-test, and draw a conclusion. **And** suppose that when you get the data back, **lo and behold** it happens to reject the null hypothesis. Given the power that your design possesses, does the result seem reliable? Or, does it seem like it might be a false-positive result?

```r
power_estimate <- mean(t_test_p_values < 0.05)
power_estimate
```

```
## [1] 0.158
```

Answer: With a calculated power of 0.138, this means that the test has only a 13.8% chance of correctly rejecting the null hypothesis when a true effect exists. Given such low power, a result where the null hypothesis is rejected may not be reliable.

3.6 Conduct a power analysis

Apply the decision rule that you wrote down in part 1 above to each of the simulations you have conducted. What proportion of your simulations have rejected your null hypothesis? This is the power that this design and testing procedure generates. After you write and execute your code, include a narrative sentence or two about what you see.

```r
t_test_rejects <- mean(t_test_p_values < 0.05)
t_test_rejects
```

```
## [1] 0.158
```

Answer: The proportion of simulations that rejected the null hypothesis (p-value < 0.05) is 0.138, which means the power of the test is 13.8%.

This low power indicates that the design and sample size are insufficient to reliably detect true effects. With such a low probability of rejecting the null hypothesis when the alternative is true, the experiment is underpowered.

3.7 Moar power!

```r
percentages_to_sample <- seq(0.1, 2, by = 0.1)  # 10% to 200%
power_results <- rep(NA, length(percentages_to_sample))

for (i in seq_along(percentages_to_sample)) {
  p_values <- rep(NA, 1000)
  sample_size <- round(percentages_to_sample[i] * nrow(d))

  for (j in 1:1000) {
    valid_sample <- FALSE
    while (!valid_sample) {
      sample_data <- d[sample(.N, sample_size, replace = TRUE)]
      group_counts <- table(sample_data$treatment)
      if (length(group_counts) == 2 && all(group_counts >= 2)) {
        valid_sample <- TRUE
      }
    }
    t_test <- t.test(bid ~ treatment, data = sample_data)
    p_values[j] <- t_test$p.value
  }

  power_results[i] <- mean(p_values < 0.05)
}

results <- data.frame(Sample_Percentage = percentages_to_sample * 100, Power = power_results)
print(results)
```

```
##    Sample_Percentage Power
## 1                 10 0.103
## 2                 20 0.201
## 3                 30 0.341
## 4                 40 0.422
## 5                 50 0.484
## 6                 60 0.577
## 7                 70 0.660
## 8                 80 0.689
## 9                 90 0.751
## 10               100 0.793
## 11               110 0.818
## 12               120 0.872
## 13               130 0.872
## 14               140 0.917
## 15               150 0.931
## 16               160 0.946
## 17               170 0.961
## 18               180 0.958
## 19               190 0.976
## 20               200 0.971
```

Answer: Increasing the sample size consistently improves the power of the test. As the percentage of the sample increases, the likelihood of rejecting the null hypothesis when the alternative is true also increases.