Consolidated Argument

Null Result Penalty Replication

Ryan McWay Emily Kurtz

2025-03-18

Direct Replication

The direct replication was sucessful. But the paper seems almost too good to be true. The point of the paper is the null results are penalized for publication. Yet all the results, even the appendix results, have huge statistically significant effects.

This is strange get the sample. They survey economists and ask them if they would publish a paper. This is measured on a sliding scale of 0 to 100. They provide each person with four of five vignettes. The authors take the vigenttes from real studies that are statistically significant and published. They keep the standard errors the same, but randomize if they shift the coefficient left in the distribution such that the effect is now statistically insignficant.

The get a sample of 480 respondents who complete four vigenettes for 1920 observations. On top of that they cross-randomize 6 other attributes of the vigenettes. Aspects such as gender, prestige, etc. could effect if the finding is publishable beyond statistical significance. This produces 48 treatment assignments using a factorial design. In practice, the authors have 40 observations per treatment assignment to identify off of – 10 respondents. Despite these small clusters, the standard errors are tiny. This makes us suspiciuos.

As part of the reproduction, we identify Table 3 and Figure 2 has presenting the main effects. Table 3 is of primary interest as it estimates the null result effect on the primary outcome of interest and the secondary outcomes. Figure 2 estimates the interaction effect of the null effect with the cross-randomized characteristics of the vigenettes. Below we represent a reproduction of the main estimate – Column 1 of Table 3.

```
# Column 1
col1 <- plm(publish ~ low + exlow + exhigh + field + phd + unilow + pval,
                data = df,
                index = c("id", "vignette"),
                model = "within")
```

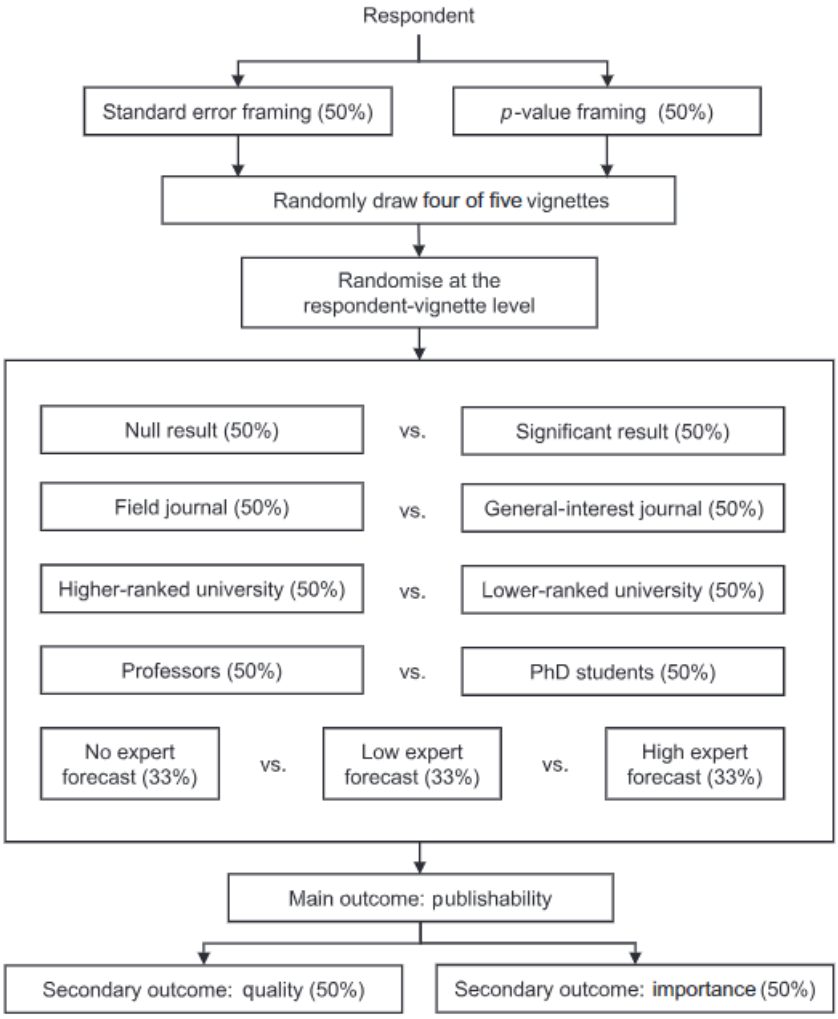


Figure 1: Factorial Design

```r
col1_se <- sqrt(diag(vcovHC(col1, type = "HC1", cluser = "id")))
    # TODO: Issue with adding clustering
df_control <- subset(df, df$low == 0)
col1_mean <- round(mean(df_control$publish), 3) # Subset for control
# Present
stargazer(col1,
        type = "text",
        keep = c(1),
        covariate.labels = c("Null result treatment"),
        se = list(col1_se),
        keep.stat = c("n", "adj.rsq"),
        model.numbers = TRUE,
        digits = 3,
        add.lines = list(c("Mean Dep. Var.", col1_mean)
        ))
```

```
================================================
                         Dependent variable:
                     ---------------------------
                                publish
------------------------------------------------
Null result treatment         -14.054***
                                (1.099)

------------------------------------------------
Mean Dep. Var.                  57.193
Observations                     1,920
Adjusted R2                     -0.070
================================================
Note:                *p<0.1; **p<0.05; ***p<0.01
```

We examine this in a couple of ways in particular.

1. Variation in the dependent variable
2. Sample composition
3. Quantile Regressions
4. Propensity score matching

The motivation for these robustness checks are to stress test the results in examining if there is potential data manipulation that ensures statistical significance. Our current results suggest that the data is unlikely to have been generated from real world data. The recommendation

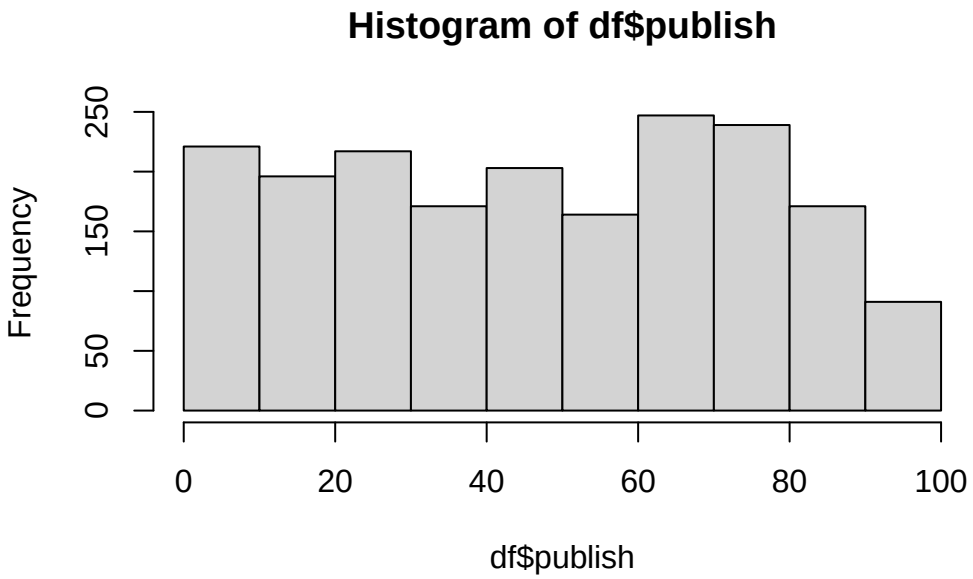
of this replication is that Chopra et al. (2023) should be replicated using new data with an independent team of researchers.

Variation in the Publishability

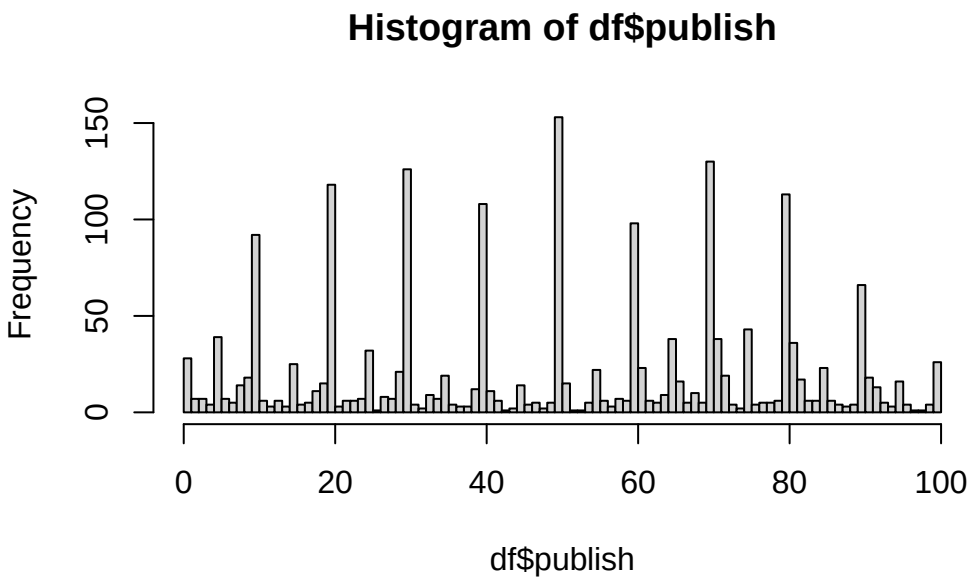
The first thing that we note is the distribution of the primary outcome of interest – publishability.

The first thing that is strange is that the outcome measure appears to be uniformly distributed. That is a bit odd. Without binning, we also see that there is some grouping around divisors of 5 along the sliding scale used by respondents.

```
# Suspiciously uniform
hist(df$publish, breaks = 10)
```

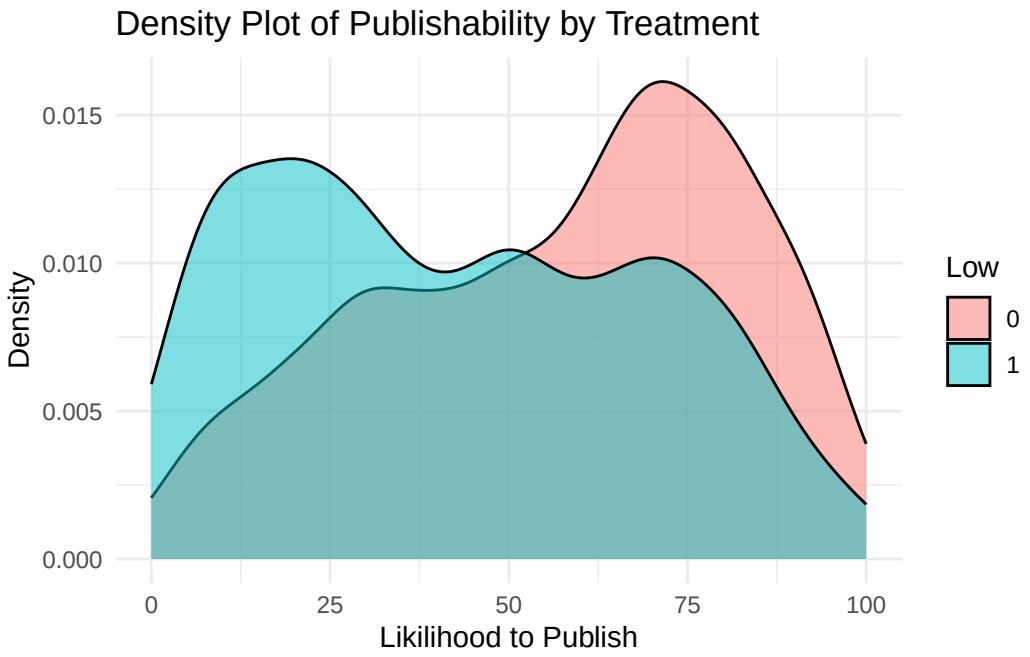



```
# Bunching around specific values
hist(df$publish, breaks = 100) # Lots of grouping on individual values
```



We notice something strange when we examine the distribution of the outcome measure when highlighting treatment assignment. Notably, the control and treatment distributions look like mirrors of one another.

```
# Publish by treatment
ggplot(df, aes(x = publish, fill = factor(low))) +
    geom_density(alpha = 0.5) +
    labs(title = "Density Plot of Publishability by Treatment",
         x = "Likilihood to Publish",
         y = "Density",
         fill = "Low") +
    theme_minimal()
```



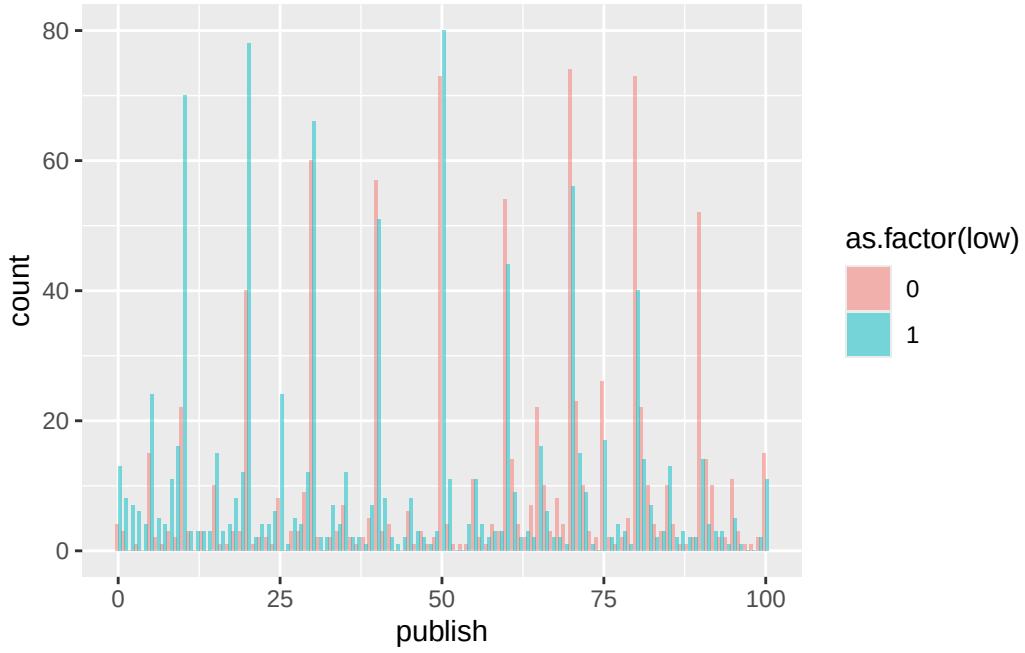
We suspect that treatment and control are the same distribution but systemtric about the middle of the range (50). In context, this is meaninful as 50 can be interperted as the threshold between publishing and not publishing the article. When we flip the control group distribution by the forumla $[publish|t_i = 0] = 100 - publish$ we find that treatment and control have the same distribution. This suggests that the data could have been generated from a random distribution rather than real data. In particular, this appears to be a Beta distribution. Using the following formula for the probability distribution function, you could reproduce the underlying data, split the sample in half, and flip the 'control' group about the range to create a reflection. With this reflection, we could produce the results from the Chopra et al. paper without collecting any data.

> The PDF for the beta distribution, for $0 \leq x \leq 1$, uses the shape parameters $\alpha, \beta > 0$ to create a power function of some variable x. The denominator is normalization to ensure total probability of 1.

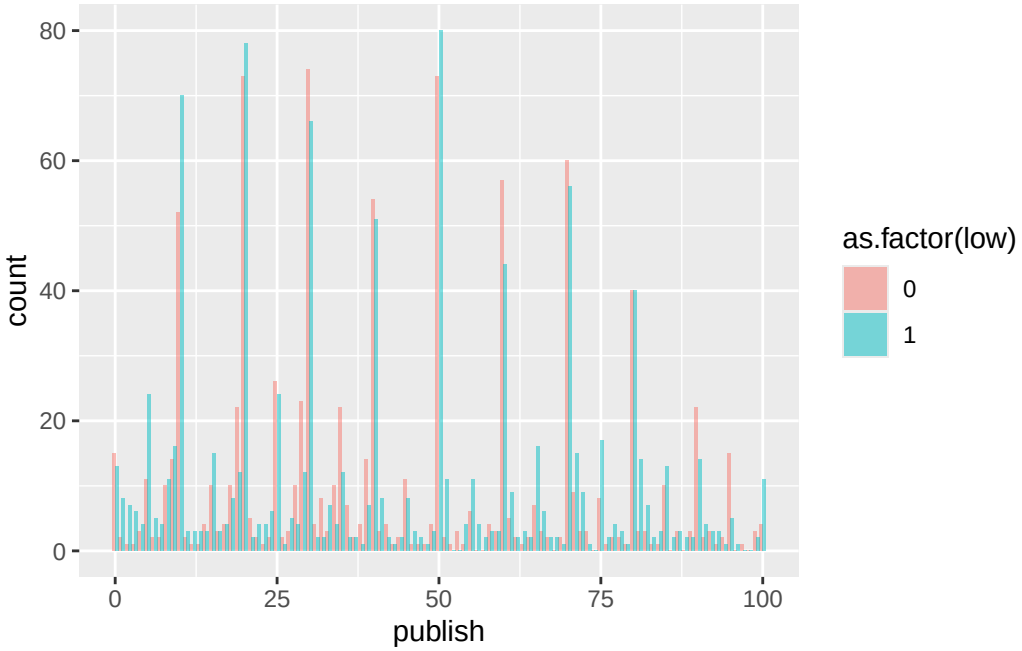
$$f(x; \alpha, \beta) = \frac{x^{\alpha-1}(1-x)^{\beta-1}}{\int_0^1 u^{\alpha-1}(1-u)^{\beta-1} du}$$

```
# Histogram overlaying control onto treatment
df2 <- data.table::copy(df)
df2[, publish := ifelse(low == 1, publish, 100 - publish)]
```

```r
# Regular historgram by treatment
ggplot(df, aes(publish, fill = as.factor(low))) +
    geom_histogram(alpha = 0.5, position = 'dodge', binwidth = 1)
```




```r
# Histogram after flip the scale (e.g., are the symettric about the average (50))
ggplot(df2, aes(publish, fill = as.factor(low))) +
    geom_histogram(alpha = 0.5, position = 'dodge', binwidth = 1)
```




```
    # This looks awfully symettric...
```

One thing that the authors could have done to make the data appear more 'realistic' is to 'jitter' the data in the distrbution and apply a heurisitic for how participants would select values. Suppose that we expect people to tend to select items that are multiple of 5s or 10s. Then I could just create this Beta distribution as a discrete function with intervals of fives. For the formula above, instead of $\int$ you could replace it with $\sum_i^n f(5i)$ to make this discrete distribution. Because this would be too neat, the authors may add some noise. Specifically, values that are not multiples of 5, as well as adding values near multiples of 5 to show human errors.

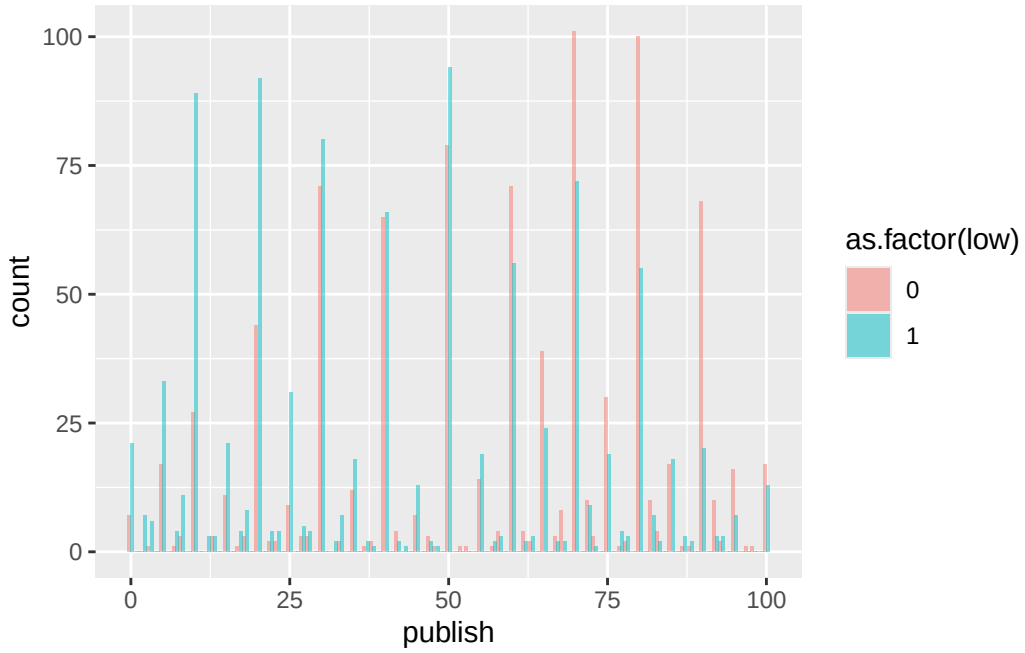
We account for this in our descriptive of the distribution by recoding values near divisors of 5 to the nearest divisor. As a bandwidth, we recode values that are 1 value away. For example, if you have a uniform distrbution from 5 to 10 you would expect the observations: $5, 6, 7, 8, 9, 10$. Using our bandwidth to recode we will now have the observations $5, 5, 7, 8, 10, 10$. In a uniform distribution, that means that rather than a $2/6$ chance of selection for divisors of 5 there is now a $4/6$ chance of divisors of 5. The increased likelihood should apply similiarly to the Beta distribution.

```
# Recode values within bandwidth
df = df[, publish := fifelse(publish %% 5 == 0, publish,
                fifelse(publish %% 5 == 1, publish - 1,
```

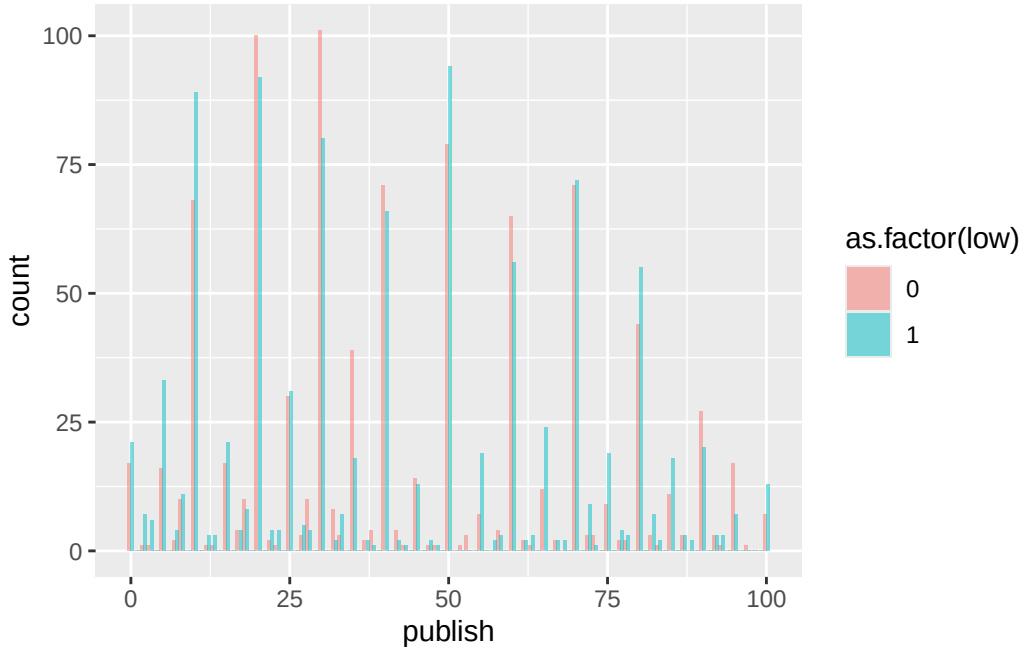
```
                fifelse(publish %% 5 == 4, publish + 1, publish)))]

# Same for overlay data set
df2 = df2[, publish := fifelse(publish %% 5 == 0, publish,
                fifelse(publish %% 5 == 1, publish - 1,
                fifelse(publish %% 5 == 4, publish + 1, publish)))]

# Replot
ggplot(df, aes(publish, fill = as.factor(low))) +
    geom_histogram(alpha = 0.5, position = 'dodge', binwidth = 1) # As presented
```




```
ggplot(df2, aes(publish, fill = as.factor(low))) +
    geom_histogram(alpha = 0.5, position = 'dodge', binwidth = 1) # With overlay
```



Content to add here Emily: - Details on distribution from other slider bars. In particular if we could get some that are from other studies predicting things on a slider from 0 to 100. - The empircal tests: kolmogorov Smirnov test

Sample Composition

- Ryan
- Two things: include the two sample selections they edit and…
- Remove observations that may not have salience of treatment (short and long duration or vigenette observations as well as 'finished == 0' observations)
- Re-estimate Table 3 effects

Quantile Regressions

- Ryan
- Derek might need to remind me of the motivation here…

```
# Quantile Regression:
    # Tau is quantile: Repeat for 0.1 to 0.9.
```

```r
taus <- seq(from = .1, to = .9, by = 0.1) # Range of quantiles
quant_all  <- rq(publish ~ low + exlow + exhigh + field + phd + unilow + pval + id + vigne
                 tau = taus,
                 data = df)

print(quant_all$coef)
```

```
               tau= 0.1      tau= 0.2      tau= 0.3     tau= 0.4      tau= 0.5
(Intercept)  21.25955414  34.314655172  41.086100861  55.57894737  64.491286452
low         -10.70700637 -15.784482759 -17.452644526 -20.72368421 -21.142043962
exlow        -0.85828025   0.314655172   2.479704797   1.80263158  -0.712934583
exhigh        1.25636943   0.426724138   2.589175892   1.60526316  -2.013836588
field         8.54140127  10.961206897  15.019680197  17.13157895  14.876434013
phd          -3.16560510  -3.935344828  -3.789667897  -4.71052632  -4.537700324
unilow       -2.75796178  -4.362068966  -4.059040590  -6.10526316  -6.399071723
pval         -2.16719745  -4.594827586  -7.879458795  -9.14473684  -7.958577809
id           -0.00477707  -0.004310345  -0.004920049  -0.01315789  -0.009282774
vignette      0.30414013   0.215517241   0.210332103   0.15789474   0.473684211
               tau= 0.6      tau= 0.7      tau= 0.8     tau= 0.9
(Intercept)  70.33793588  74.030334418  82.626030980 92.061966771
low         -18.85396677 -13.811482687 -11.293099980 -8.951728783
exlow        -0.90007021  -0.639686298  -3.082076041 -2.583071396
exhigh       -0.74210157  -1.529076650  -3.642526655 -2.575886843
field        14.58436696  12.861867416  11.492858580  8.572743601
phd          -4.78212029  -2.727360166  -4.136189901 -4.852716659
unilow       -4.49660660  -2.634581237  -3.862804265 -1.740682533
pval         -6.50994617  -5.573838414  -4.721182861 -3.515491693
id           -0.01193541  -0.004217224  -0.004023335 -0.004490346
vignette      0.29042827  -0.041876295   0.427881714 -0.439155815
```

```r
print(quant_all$coef[2,])
```

```
  tau= 0.1   tau= 0.2   tau= 0.3   tau= 0.4   tau= 0.5   tau= 0.6   tau= 0.7
-10.707006 -15.784483 -17.452645 -20.723684 -21.142044 -18.853967 -13.811483
  tau= 0.8   tau= 0.9
-11.293100  -8.951729
```

```r
# NOTE: So there is variation over the quantiles... good sign
```

```
# q01_se <- sqrt(diag(vcovHC(q_05, type = "HC1", cluser = "id")))
#     # TODO: Issue with quantile regression with FE
#     # TODO: Do for other values

# # Present
# stargazer(col1,
#         type = "text",
#         keep = c(1),
#         covariate.labels = c("Null result treatment"),
#         se = list(col1_se),
#         keep.stat = c("n", "adj.rsq"),
#         model.numbers = TRUE,
#         digits = 3,
#         add.lines = list(c("Mean Dep. Var.", col1_mean)
#         ))
# # Summarize the results
# summary(quantile_reg)
```

```
# TODO: Need to recover the SE to create CI for the plots.


# Plot the quantile regressions
# plot_models(ols, quant_reg_med, quant_reg_first, quant_reg_last,
#            show.values = TRUE,
#            m.labels = c("OLS", "Median", "10th percentile",
#                         "95th percentile",
#                         legend.title = "Model")
#            )
```

Propensity Score Matching

- Ryan/Derek...
- Create Propensity scores with logit
- Do matching
- Estimated effect on matched pairs