Calculating Adjusted p-Values From EdSurvey Results

Developed by Paul Bailey, Michael Lee, and Ting Zhang[*†]

April 30, 2018

Introduction

This vignette will describe the basics of adjusting p-values for analyses in the `EdSurvey` package and is structured as follows:

- Background of Multiple Comparisons of National Assessment of Educational Progress (NAEP) Data
- Calculating Multiple Comparisons in EdSurvey
- Adjusting p-values From Multiple Sources

Background of Multiple Comparisons of NAEP Data

When making groups or families of comparisons in a single analysis, such as comparing White students with minority student groups in terms of test scores, the probability of finding significance by chance for at least one comparison increases with the family size or the number of comparisons. Multiple methods exist to adjust p-values to hold the significance level for a set of comparisons at a particular level (e.g., 0.05), and such adjustments are called multiple comparison procedures. The (NAEP) employs two procedures: the Benjamini-Hochberg false discovery rate (FDR) procedure (Benjamini & Hochberg, 1995) and the Bonferroni procedure. The Bonferroni procedure was used prior to the 1996 assessment. Thereafter, NAEP has used the FDR procedure. A detailed explanation of the NAEP multiple comparison procedures can be found at Comparison of Multiple Groups.

Typically, the number of comparisons is determined as the number of possible statistical tests in a single analysis. However, in NAEP reports, when comparing multiple years and multiple jurisdictions (e.g., multiple states versus the United States as a whole), usually neither the number of years nor the number of jurisdictions counts toward the number of comparisons.

The next section illustrates how to adjust the p-values using the Bonferroni and FDR procedures through R's `p.adjust` function.

Calculating Multiple Comparisons in EdSurvey

Before processing begins, load the `EdSurvey` package and the NAEP data to be analyzed, in this case the Primer Data included in the `NAEPprimer` package:

```
library(EdSurvey)
sdf <- readNAEP(system.file("extdata/data", "M36NT2PM.dat", package = "NAEPprimer"))
```

Once read in, a linear model is analyzed by the `lm.sdf` function—in this case, `dsex`, `b017451`, the five plausible values for `composite`, and the full sample weight `origwt`.

```
lm1 <- lm.sdf(formula = composite ~ dsex + b003501 + b003601, data = sdf)
summary(lm1)$coefmat
```

*This publication was prepared for NCES under Contract No. ED-IES-12-D-0002 with the American Institutes for Research. Mention of trade names, commercial products, or organizations does not imply endorsement by the U.S. Government.

†The authors would like to thank Dan Sherman for reviewing this document.

Table 1: Coefficients

	coef	se	t	dof	Pr(>|t|)
(Intercept)	262.524094	1.3229499	198.4384271	43.30064	0.0000000
dsexFemale	-1.518125	0.9136723	-1.6615637	68.46417	0.1011731
b003501Graduated H.S.	4.081660	1.5789000	2.5851287	41.10442	0.0133794
b003501Some ed after H.S.	15.030180	1.4053437	10.6950208	45.41276	0.0000000
b003501I Don't Know	-1.591764	1.7993156	-0.8846496	34.72285	0.3824307
b003601Graduated H.S.	2.897892	1.6544505	1.7515734	44.98221	0.0866633
b003601Some ed after H.S.	9.154892	1.8554737	4.9339916	25.78984	0.0000408
b003601I Don't Know	-4.120843	1.5267240	-2.6991407	37.56060	0.0103599

Note

The `EdSurvey` package produces p-values based on the assumption that tests are independent and unassociated with each other; yet this assumption is not always valid. Several possible methods have been developed for dealing with the multiple hypothesis testing problem.

The p-values for variables run in `lm1` can be corrected for multiple testing. Notice that the only p-values adjusted in this example are in rows six, seven, and eight of the coefficients in `lm1`:

```
summary(lm1)$coefmat[6:8,]
```

Table 2: Rows Six, Seven, and Eight

	coef	se	t	dof	Pr(>|t|)
b003601Graduated H.S.	2.897892	1.654451	1.751573	44.98221	0.0866633
b003601Some ed after H.S.	9.154892	1.855474	4.933992	25.78984	0.0000408
b003601I Don't Know	-4.120843	1.526724	-2.699141	37.56060	0.0103599

Column 5 has the p-values in it, so that column is selected. Here the Benjamini and Hochberg (1995) FDR adjustment is used in the argument `method = "BH"`. The results show that all the adjusted p-values are one:

```
p.adjust(p = lm1$coefmat[6:8, 5], method = "BH")
```

Table 3: BH-Adjusted Rows Six, Seven, and Eight

coef	se	t	dof	Pr(>|t|)	Adjusted Pr(>|t|)
2.897892	1.654451	1.751573	44.98221	0.0866633	0.0866633
9.154892	1.855474	4.933992	25.78984	0.0000408	0.0001225
-4.120843	1.526724	-2.699141	37.56060	0.0103599	0.0155399

The next example adjusts the same p-values using the Bonferroni adjustment with `method="bonferroni"`. The adjusted p-values are, again, all one:

```
p.adjust(p = lm1$coefmat[6:8, 5], method = "bonferroni")
```

```
## [1] 0.2599899019 0.0001224987 0.0310798488
```

The coefficients matrix also can be overwritten by selecting the same vector in the `lm1` linear regression object:

```r
# Note 3:6 are the rows we want (those regarding b017451)
# and column 5 is the p-value column
lm1$coefmat[6:8, 5] <- p.adjust(lm1$coefmat[6:8, 5], method = "bonferroni")
summary(lm1)
```

```
##
## Formula: composite ~ dsex + b003501 + b003601
##
## jrrIMax: 1
## Weight variable: 'origwt'
## Variance method: jackknife
## JK replicates: 62
## full data n: 17606
## n used: 8988
##
## Coefficients:
##                               coef        se         t    dof  Pr(>|t|)
## (Intercept)                262.52409   1.32295 198.43843 43.301 < 2.2e-16 ***
## dsexFemale                  -1.51812   0.91367  -1.66156 68.464 0.1011731
## b003501Graduated H.S.        4.08166   1.57890   2.58513 41.104 0.0133794 *
## b003501Some ed after H.S.   15.03018   1.40534  10.69502 45.413 5.418e-14 ***
## b003501I Don't Know         -1.59176   1.79932  -0.88465 34.723 0.3824307
## b003601Graduated H.S.        2.89789   1.65445   1.75157 44.982 0.2599899
## b003601Some ed after H.S.    9.15489   1.85547   4.93399 25.790 0.0001225 ***
## b003601I Don't Know         -4.12084   1.52672  -2.69914 37.561 0.0310798 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Multiple R-squared: 0.0758
```

Adjusting *p*-Values From Multiple Sources

Sometimes several values need to be adjusted at once. In these cases, the `p.adjust` function must be called with all the *p*-values the researcher wishes to adjust together.

For example, if one wishes to adjust values from two regressions and an additional value from another test, all these *p*-values must be put into a single vector and adjusted as a set. Therefore, *p*-value adjustments called on smaller portions of regressions/tests independently may return incorrect adjusted *p*-values and could result in an incorrect inference.

In this example, the coefficients from `b003501` and `b003601`—each of independent regressions—as well as another *p*-value of 0.02 are adjusted.

```r
lm2a <- lm.sdf(formula = composite ~ dsex + b003501, data = sdf)
lm2b <- lm.sdf(formula = composite ~ dsex + b003601, data = sdf)

# pvalues data.frame with missing values
# values of coef that are not in this initial call but will be added
pvalues <- data.frame(source=c(rep("lm2a",3), rep("lm2b",3), "otherp"),
                      coef=rep("",7),
                      p=rep(NA,7),
                      stringsAsFactors=FALSE)
```

This code is careful to note where the values came from to help avoid transcription errors. The `pvalues`

object is then populated using p-values and coefficients from the `lm2a` and `lm2b` linear regression objects, rows one through three and four through six for each, respectively.

```
# load in values from lm2a
lm2aCoef <- summary(lm2a)$coefmat
pvalues$p[1:3] <- lm2aCoef[3:5,5]
pvalues$coef[1:3] <- row.names(lm2aCoef)[3:5]

# load in values from lm2b
lm2bCoef <- summary(lm2b)$coefmat
pvalues$p[4:6] <- lm2bCoef[3:5,5]
pvalues$coef[4:6] <- row.names(lm2aCoef)[3:5]
```

The additional p-value due for adjustment is included in row seven:

```
# load in other p-value
pvalues$p[7] <- 0.02
colnames(pvalues)[3] "Pr(>|t|)"

# check matrix
pvalues
```

Table 4: Unadjusted p-values

source	coef	Pr(>|t|)
lm2a	b003501Graduated H.S.	0.0000088
lm2a	b003501Some ed after H.S.	0.0000000
lm2a	b003501I Don't Know	0.0295389
lm2b	b003501Graduated H.S.	0.0000144
lm2b	b003501Some ed after H.S.	0.0000000
lm2b	b003501I Don't Know	0.0013006
otherp		0.0200000

Now that the aforementioned p-values are included in the same vector, they are adjusted via `p.adjust` using the Benjamini and Hochberg method:

```
pvalues[,"Adjusted Pr(>|t|)"] <- p.adjust(p = pvalues[,"Pr(>|t|)"], method = "BH")
pvalues
```

Table 5: Adjusted p-values

source	coef	Pr(>|t|)	Adjusted Pr(>|t|)
lm2a	b003501Graduated H.S.	0.0000088	0.0000205
lm2a	b003501Some ed after H.S.	0.0000000	0.0000000
lm2a	b003501I Don't Know	0.0295389	0.0295389
lm2b	b003501Graduated H.S.	0.0000144	0.0000253
lm2b	b003501Some ed after H.S.	0.0000000	0.0000000
lm2b	b003501I Don't Know	0.0013006	0.0018208
otherp		0.0200000	0.0233333

Reference

Benjamini, Y., & Hochberg, Y. (1995). Controlling the false discovery rate: A practical and powerful approach to multiple testing. *Journal of the Royal Statistical Society, Series B (Methodological)*, *57*(1), 289–300.