

BEYOND TWO VARIABLES AND NULL HYPOTHESIS SIGNIFICANCE TESTING

This book assumes you have a working knowledge of R (Version 3.6.0; R Core Team, 2019) and RStudio (Version 1.2.1335; RStudio Team, 2018). I recommend readers who are new to R refer to the first volume (Rasco, 2020a) or similar resources to gain a foundation (e.g., importing and exporting files, creating objects, using functions) before moving forward. The first volume addresses downloading the programs, using projects folders in RStudio, writing object-oriented code, obtaining basic statistics (e.g., mean, median, standard deviation) and graphs (e.g., scatterplots, bar charts), and performing bivariate analyses (e.g., independent-samples t test, one-way analysis of variance, and bivariate correlation and regression). Consequently, I recommend reviewing these topics if you are not familiar with performing these tasks in R. Assuming you have the requisite knowledge, this book builds on this foundation and discusses approaches beyond null hypothesis significance testing and how to include additional variables as covariates (e.g., partial correlation), multiple predictors (e.g., multiple regression), and multiple outcomes (e.g., multivariate analysis of variance). To begin, we will discuss alternatives to traditional statistical significance tests, including confidence intervals, effect sizes, and meta-analysis.

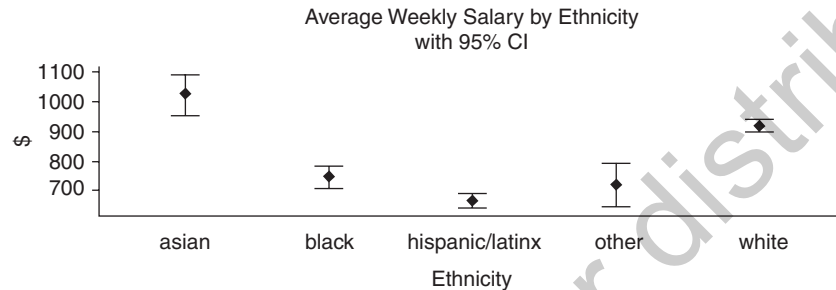
CONFIDENCE INTERVALS

Researchers are increasingly concerned about issues with null hypothesis significance testing (see Warner, 2020b, for discussion). Confidence intervals help to address a couple of these issues as they more intuitively remind the reader about the variability in the data and how findings could be different in future research. These confidence intervals can be presented visually using error bars (see Figure 1.1 from Rasco, 2020a), although the interpretation of these error bars is slightly difficult at times.

The error bars presented in Figure 1.1 provide a likely range for means obtained by ethnic group (i.e., Asian, Black, Hispanic/Latin(x), other, White) if another study was conducted on a similar population. However, the statistical tests we conduct are often focused on the differences between ethnicities. If two error bars in Figure 1.1 do not overlap, the interpretation of the graph in relation to the statistical tests is easy: A two-tailed test of the difference between those ethnicities will yield a significant

difference. In contrast, if the two error bars overlap, it is still possible there is a significant difference between the ethnicities because the statistical test is comparable with a confidence interval of the difference between the ethnicities, specifically whether the confidence interval of the difference includes zero.

Figure 1.1



You can create a chart that is more consistent with the tests conducted by graphing the confidence intervals for the differences. For example, post hoc comparisons for the analysis of variance (ANOVA) results could be saved to an object (*TukeyOut*) using the *TukeyHSD* function, and the columns and rows needed from the output could be saved to a new data frame (*ChartEx*) using the *data.frame* function. Then, the *ggplot2* package (Wickham, 2016) can be used to graph the confidence intervals from the output provided by the *TukeyHSD* function.

In Figure 1.2, you can see an example building on the analyses from Chapter 13 in the first volume (Rasco, 2020a). In this example, the first four rows (*[1:4, 1:3]*) of the first three columns (*[1:4, 1:3]*) from the *TukeyHSD* output (*TukeyOut*) are saved to a new data frame (*ChartEx*), and a bar chart is created to visualize the differences and confidence intervals for all of the comparisons between each of the ethnic groups and the Asian ethnicity. These confidence intervals show individuals who identify as Asian made more money than each of the other groups (e.g., White, Black) because the confidence intervals for the differences do not contain zero and all of the differences are negative.

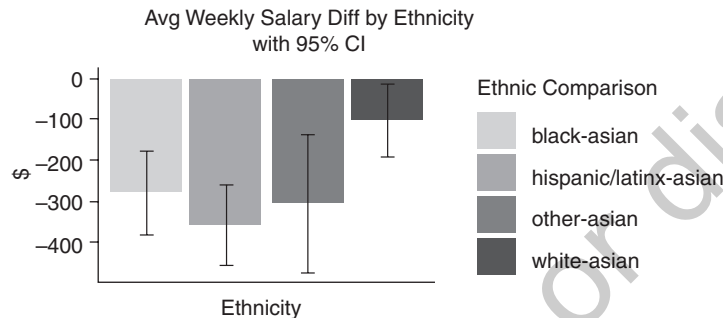
```
TukeyOut <- TukeyHSD(ezResult$aov)
ChartEx <- data.frame(EthnicComp=rownames(TukeyOut$race_
  ethnicity[1:4,]),
  TukeyOut$race_ethnicity[1:4,1:3], row.names = NULL)
ggplot(ChartEx, aes(x=EthnicComp, y=diff, fill=EthnicComp)) +
  geom_bar(stat="identity") +
  geom_errorbar(aes(ymin=lwr, ymax=upr), width=.2) +
  scale_fill_brewer("Ethnic Comparison",
    palette="Dark2") +
  theme_classic()+
  labs(title="Avg Weekly Salary Diff by Ethnicity",
```

```

    subtitle="with 95% CI", x="Ethnicity",
    y="$")+
  theme(axis.ticks.x=element_blank(),
        axis.text.x = element_blank(),
        plot.title=element_text(hjust=.5),
        plot.subtitle=element_text(hjust=.5))

```

Figure 1.2



EFFECT SIZE

Measures of effect size provide another option that helps us address issues with traditional null hypothesis testing. Specifically, the null hypothesis is basically always wrong. For example, in a one-sample t test, the population and sample are rarely (if ever) perfectly equal, which is the null hypothesis. As a result of this inherent difference, we can reject the null hypothesis in almost all cases with a large enough sample size.

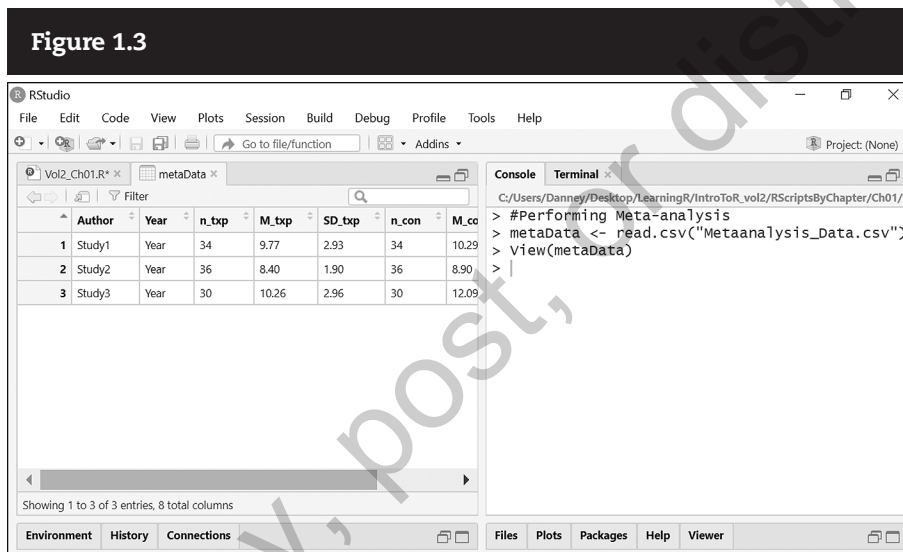
Effect size measures remove the influence of the sample size on the findings. In the case of the one-sample t test, Cohen's d is a commonly used effect size measure, and this measure focuses on the difference between the sample and the population, accounts for the variability in the sample, and ignores the sample size. Thus, we obtain a ratio of the difference between the sample and the population in relation to the unique (i.e., unexplained) variability in the sample. If the difference is relatively large compared to the variability, the effect size (i.e., Cohen's d) will be relatively large. In contrast, if the difference is small compared to the variability, the effect size will be small. Consequently, this measure maintains the continuous nature of the original sample and population measures.

Fortunately, in R, there are many options for effect size. The t_apa function from the *apa* package (Gromer, 2019), for example, provides the effect size measure for t tests in the American Psychological Association (APA)-style results. Additionally, the *compute.es* package (del Re, 2013) and similar packages (e.g., *effsize*; Torchiano, 2019) can compute commonly reported effect size estimates. Finally, the output provided by a statistical test (e.g., one-way analysis of variance) can be used to manually compute estimates of effect size (e.g., η^2).

META-ANALYSIS

Meta-analyses allow us to look at a group of previously conducted studies together. As an example, import the *Metaanalysis_Data.csv* file: `metaData <- read.csv("Metaanalysis_Data.csv")`. This file contains the information presented in Warner (2020b; adapted from Ried, 2008). You can *View* the *metaData* object to learn more about the structure of the data used for meta-analyses (Figure 1.3). It contains eight columns for each study: author name, year, descriptive statistics for the experimental or treatment group (sample size [*n_txp*], mean [*M_txp*], and standard deviation [*SD_txp*]), and descriptive statistics for the control group (*n_con*, *M_con*, *SD_con*).

Figure 1.3



The screenshot shows the RStudio interface. The main window displays a data table with 8 columns: Author, Year, n_txp, M_txp, SD_txp, n_con, M_con, and SD_con. The data is as follows:

	Author	Year	n_txp	M_txp	SD_txp	n_con	M_con	SD_con
1	Study1	Year	34	9.77	2.93	34	10.29	
2	Study2	Year	36	8.40	1.90	36	8.90	
3	Study3	Year	30	10.26	2.96	30	12.09	

The console window shows the following R code and output:

```
C:/Users/Danney/Desktop/LearningR/IntroToR_vol2/RScriptsByChapter/Ch01/
> #Performing Meta-analysis
> metaData <- read.csv("Metaanalysis_Data.csv")
> View(metaData)
> |
```

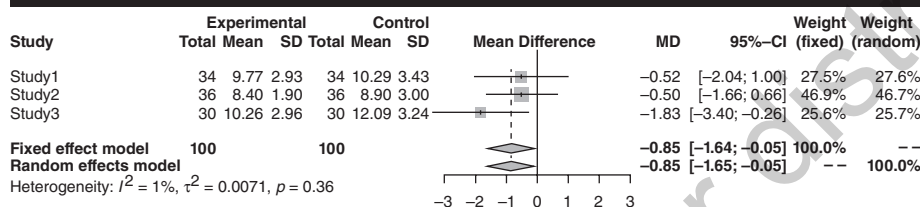
Next, install (*install.packages*) and attach (*library*) the *meta* package (Schwarzer, 2007), and use the *metacont* function to create the meta-analysis output. This function uses several arguments. For the experimental group, you need sample size (*n.e*), mean (*mean.e*), and standard deviation (*sd.e*). You need these values for the control group too: *n.c*, *mean.c*, and *sd.c*, respectively. Additionally, the output is easier to read if you provide a label, commonly the first author or two, for each study (*studlab*).

```
metaOut <- metacont(n.e=n_txp, mean.e=M_txp, sd.e=SD_txp,
n.c=n_con, mean.c=M_con, sd.c=SD_con, data=metaData,
studlab=Author, method.smd="Cohen", pooledvar=TRUE)
```

Once the meta-analysis output object is created, you can use the *forest* function to present the results: `forest(metaOut, digits.sd=2)`. This output notes the studies included, provides the original data (i.e., sample size, mean, standard deviation), and

includes the mean difference (“MD”) and confidence interval for each study. Further, it provides a confidence interval for the effect across the three studies [-1.64, -0.05]. As this interval does not contain zero, it would suggest that symptoms are lower in the treatment conditions compared to the control conditions across the three studies. The meta-analysis method used Cohen’s *d* (*method.smd* = “Cohen”) as the effect size measure, and the average difference across the three studies was -0.85. Consequently, it appears therapy reduces symptoms by 0.85 standard deviations.

Figure 1.4



Chapter 1 Summary of Key Functions (AKA: Function Cheat Sheet)

Function Call	Package	Description
setwd	Included in base	Sets working directory
read.csv	Included in utils	Assigns data set from CSV file to data frame
ezANOVA	Ez	Calculates ANOVA results
TukeyHSD	Included in stats	Performs pairwise post hoc analyses for ANOVA
group_by	Dplyr	Groups table by categorical variable(s)
summarize	Dplyr	Summarizes variables by group if using group_by
geom_bar	ggplot2	Creates bar graph
geom_errorbar	ggplot2	Generates error bars
geom_point	ggplot2	Produces scatterplot or layers point on plot
data.frame	Included in base	Creates a new data frame
metacont	meta	Calculates effect estimates for meta-analyses
forest	meta	Draws forest plot for object from metacont