

# *SOC 4015/5050: Lecture 08 Functions*

*Christopher Prener, Ph.D.*

*Fall 2018*

## *Packages*

- car
- effsize
- ggribes
- ggstatsplot
- pwr
- stats
- tidyverse
  - broom
  - ggplot2
  - readr

## *Reading and Writing Data*

### *Reading Data*

```
readr::read_csv(path = filePath)
```

You need to use the here package to build your file paths correctly.

### *Writing Data*

```
readr::write_csv(dataFrame, path = filePath)
```

## *Tidy Output*

```
broom::tidy(testFunction)
```

## *Saving Plots*

```
ggplot2::ggsave(filename, dpi = val)
```

This use of ggsave will save the plot you have created most recently.

*Plots for Mean Difference**Box Plot*

```
ggplot2::geom_boxplot(mapping = aes(aesthetic))
```

*Violin Plot*

```
ggplot2::geom_violin(mapping = aes(aesthetic))
```

*Violin Plot with Mean Points*

```
ggplot2::geom_violin(mapping = aes(aesthetic)) +  
ggplot2::stat_summary(fun.y = mean, geom = "point")
```

You need to set the base aesthetic mapping in your initial `ggplot()` call.

*Ridge Plot*

```
ggridges::geom_density_ridges(mapping = aes(aesthetic))
```

*Ridge Plot with Transparent Fill*

```
ggridges::geom_density_ridges(mapping = aes(aesthetic),  
  alpha = val)
```

*Stats Plot*

```
ggstatsplot::ggbetweenstats(data = dataframe,  
  x = xvar, y = yvar, effsize.type = "biased",  
  plot.type = plotType)
```

You *do not* need to call the `ggplot()` function first! Valid `plot.type` values are "violin", "box", and "boxviolin".

*Levene's Test*

```
car::leveneTest(yVar ~ xVar, data = dataframe)
```

*One-Sample T Test*

```
stats::t.test(dataFrame$yVar, mu = val)
```

*Two-Sample (Independent) T Test*

```
stats::t.test(dataFrame$yVar ~ dataFrame$xVar,
  var.equal = FALSE)
```

Do not forget to adjust the value of `var.equal` based on the findings of the Levene's test.

*Reshaping Data**Wide to Long*

```
tidyr::gather(dataFrame, key, value, ...)
```

*Long to Wide*

```
tidyr::spread(dataFrame, key, value)
```

*Dependent T Test*

```
stats::t.test(dataFrame$y1, dataFrame$y2, paired = TRUE)
```

*Cohen's d**Independent Observations*

```
effsize::cohen.d(dataFrame$yVar ~ dataFrame$xVar,
  pooled = TRUE, paired = FALSE)
```

*Dependent Observations*

```
effsize::cohen.d(dataFrame$y1, dataFrame$y2,
  paired = TRUE)
```

*Power Analysis*

```
pwr::pwr.t.test(d = val, power = val, sig.level = val,
  type = type, alternative = "two.sided")
```

Valid type values are "one.sample", "two.sample", and "paired".