CSP 519 Lab 7: Mediation

Wednesday Bushong

4/14/2017

Today we're going to learn about mediation analyses!

The Data

First, we're going to load the data:

```
library(foreign)
library(ggplot2)
```

```
## Warning: package 'ggplot2' was built under R version 3.3.2
```

```
d <- read.spss("Data.sav", to.data.frame = TRUE)
head(d)
```

```
##         se narciss    mental
## 1 2.900000      19 -2.396314
## 2 3.800000      17  4.481908
## 3 4.000000      21  2.646707
## 4 2.444444       9 -1.960831
## 5 4.000000      24  1.922139
## 6 3.800000      23  3.701790
```

Variables of interest:

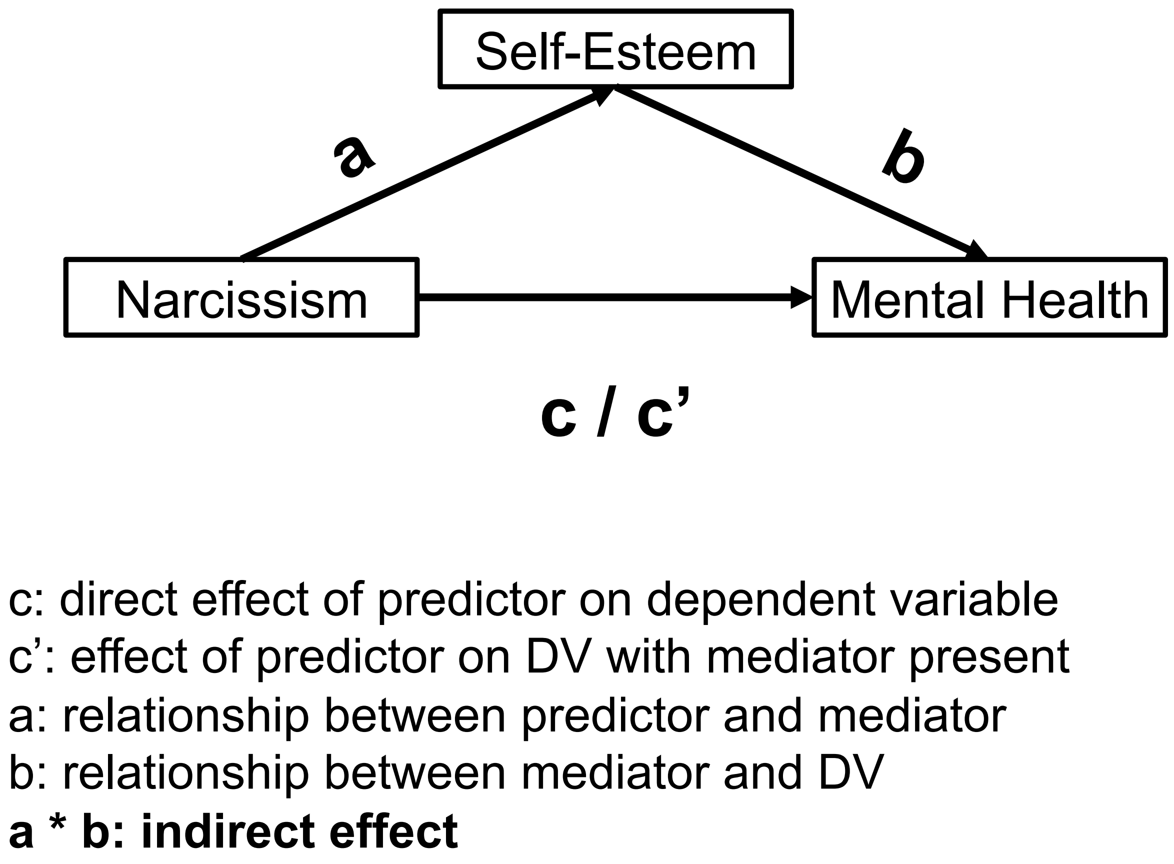
- IV: Narcissism (`d$narciss`)
- IV: Self-Esteem (`d$se`)
- DV: Mental Health (`d$mental`)

In this particular dataset, we are interested in whether self-esteem acts as a *mediator* between self-esteem and mental health. In order to do this, we'll analyze our data using a mediation analysis. See the figure below for how we'll go about testing a mediation effect:

Step 1: Test the direct effect ("c")

To test the direct effect, we'll predict mental health from narcissism.

```
c <- lm(mental ~ narciss, d)
summary(c)
```



c: direct effect of predictor on dependent variable
c': effect of predictor on DV with mediator present
a: relationship between predictor and mediator
b: relationship between mediator and DV
a * b: indirect effect

Figure 1: Relevant variables and relationships in mediation analysis.

```
##
## Call:
## lm(formula = mental ~ narciss, data = d)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -11.2332  -1.8665   0.5853   2.2413   5.5016
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -1.54889    0.62195  -2.490  0.01370 *
## narciss      0.09323    0.03450   2.703  0.00756 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.205 on 174 degrees of freedom
## Multiple R-squared:  0.04029,    Adjusted R-squared:  0.03478
## F-statistic: 7.305 on 1 and 174 DF,  p-value: 0.007558
```

There's a significant effect of narcissism on mental health (B = 0.093, p = 0.008).

Step 2: Test the relationship between predictor and mediator ("a")

```r
a <- lm(se ~ narciss, d)
summary(a)
```

```
##
## Call:
## lm(formula = se ~ narciss, data = d)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.7230 -0.3469  0.0061  0.3466  1.1353
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 2.689542   0.096175  27.965  < 2e-16 ***
## narciss     0.029187   0.005334   5.471 1.53e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.4957 on 174 degrees of freedom
## Multiple R-squared:  0.1468, Adjusted R-squared:  0.1419
## F-statistic: 29.94 on 1 and 174 DF,  p-value: 1.534e-07
```

Narcissism and self-esteem are significantly correlated (B = 0.029, p = 0).

Step 3: Test the relationship between mediator and DV, controlling for predictor ("b")

```r
b <- lm(mental ~ narciss + se, d)
summary(b)
```

```
##
## Call:
## lm(formula = mental ~ narciss + se, data = d)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -6.4353 -1.3361  0.0637  1.6748  6.1246
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -13.85831    1.03292 -13.417   <2e-16 ***
## narciss      -0.04035    0.02646  -1.525    0.129
## se            4.57678    0.34735  13.176   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.271 on 173 degrees of freedom
## Multiple R-squared:  0.521,  Adjusted R-squared:  0.5155
## F-statistic: 94.08 on 2 and 173 DF,  p-value: < 2.2e-16
```

Self-esteem has a significant effect on mental health after controlling for narcissism ($B = 4.577$, $p = 0$).

Step 4: Test the significance of the indirect effect ("a * b")

In order to test the significance of the indirect effect, we will need to *bootstrap* confidence intervals on the product between a and b in our mediation model above. Here is the series of steps we'll be taking to do this:

1. Create a 'new' dataset that is *sampled* from the old dataset. That is, we want a dataset that is exactly as many rows as our original dataset that is built up of samples from that data. In order to do this, we will sample rows from our data frame *with replacement.*
2. Compute a, b, and a*b for the sampled dataset.
3. Repeat Steps 1-2 *R* times (usually 1000).
4. Obtain the 2.5% and 97.5% quantiles from the resulting distribution; this will give us our 95% confidence intervals.
5. See if the confidence intervals from (4) overlap with zero.

Steps 1 & 2: Compute Indirect Effect For a Resampled Dataset

First, we will need a way to resample our data and compute the indirect effect. For this, I've written a function for you to do this! Don't worry too much about some of the details – they're just to make this function more general. Just notice that essentially what I'm doing is resampling the data and computing a, b, and their product.

```r
compute.indirect.effect <- function(data, predictor, mediator, dv, r = TRUE) {
  # Sample rows of data
  sample <- sample(nrow(data), replace = r)
  # Create relevant formulas (don't worry about this, this is just to make the function more general)
  a.formula <- formula(paste(mediator, "~", predictor))
  b.formula <- formula(paste(dv, "~", predictor, "+", mediator))
  # Fit the 'a' and 'b' models
  a.model <- lm(a.formula, data[sample, ])
  b.model <- lm(b.formula, data[sample, ])
  # Extract the B's from these models
  a <- summary(a.model)$coefficients[predictor, "Estimate"]
  b <- summary(b.model)$coefficients[mediator, "Estimate"]
  # Compute indirect effect term
  indirect.effect <- a * b
  return(indirect.effect)
}
```

Let's see how our basic indirect effect computation works. If we run the function with `r = FALSE`, our data sample will be identical to our original data and we will get the same result every time. This is the empirical indirect effect in our data.

```r
compute.indirect.effect(d, "narciss", "se", "mental", r = FALSE)
```

```
## [1] 0.1335803
```

```r
compute.indirect.effect(d, "narciss", "se", "mental", r = FALSE)
```

```
## [1] 0.1335803
```

```r
compute.indirect.effect(d, "narciss", "se", "mental", r = FALSE)
```

```
## [1] 0.1335803
```

```r
compute.indirect.effect(d, "narciss", "se", "mental", r = FALSE)
```

```
## [1] 0.1335803
```

If we run the function with the default setting `r = TRUE`, then we will be sampling new datasets and will come up with a different indirect effect result each time:

```r
compute.indirect.effect(d, "narciss", "se", "mental")
```

```
## [1] 0.09416616
```

```r
compute.indirect.effect(d, "narciss", "se", "mental")
```

```
## [1] 0.09484945
```

```r
compute.indirect.effect(d, "narciss", "se", "mental")
```

```
## [1] 0.1348081
```

```r
compute.indirect.effect(d, "narciss", "se", "mental")
```

```
## [1] 0.09697712
```

Step 3: Repeat 1000 times

In order to repeat the `compute.indirect.effect` function 1000 times, we will use the `replicate` function in R:

```r
bootstrapped.samples <- replicate(1000, compute.indirect.effect(d, "narciss", "se", "mental"))
head(bootstrapped.samples)
```

```
## [1] 0.17044300 0.10618511 0.11538258 0.13897927 0.12964641 0.08729799
```

Now we have a vector of bootstrap-sampled indirect effects.

Steps 4 & 5: Compute confidence intervals and check overlap

In order to compute the 95% confidence intervals, we will use the `quantile` function.

```r
confidence.intervals <- quantile(bootstrapped.samples, probs = c(0.025, 0.975))
confidence.intervals
```

```
##       2.5%      97.5%
## 0.08043529 0.19141225
```

The confidence intervals don't overlap with zero, so we have evidence for an indirect effect (i.e., that narcissism has an indirect effect on mental health as mediated by self-esteem). For reporting this analysis in your homework or in a paper, you would want to report the empirical indirect effect first. We did this above but I'm repeating it here now:

```r
original.indirect.effect <- compute.indirect.effect(d, "narciss", "se", "mental", r = FALSE)
```

Then, you would report the confidence intervals from your bootstrapped analysis. E.g., "The indirect effect of narcissism on mental health as mediated by self-esteem was significant (B = 0.134, confidence intervals = [0.08, 0.19])".

Plotting Indirect Effects

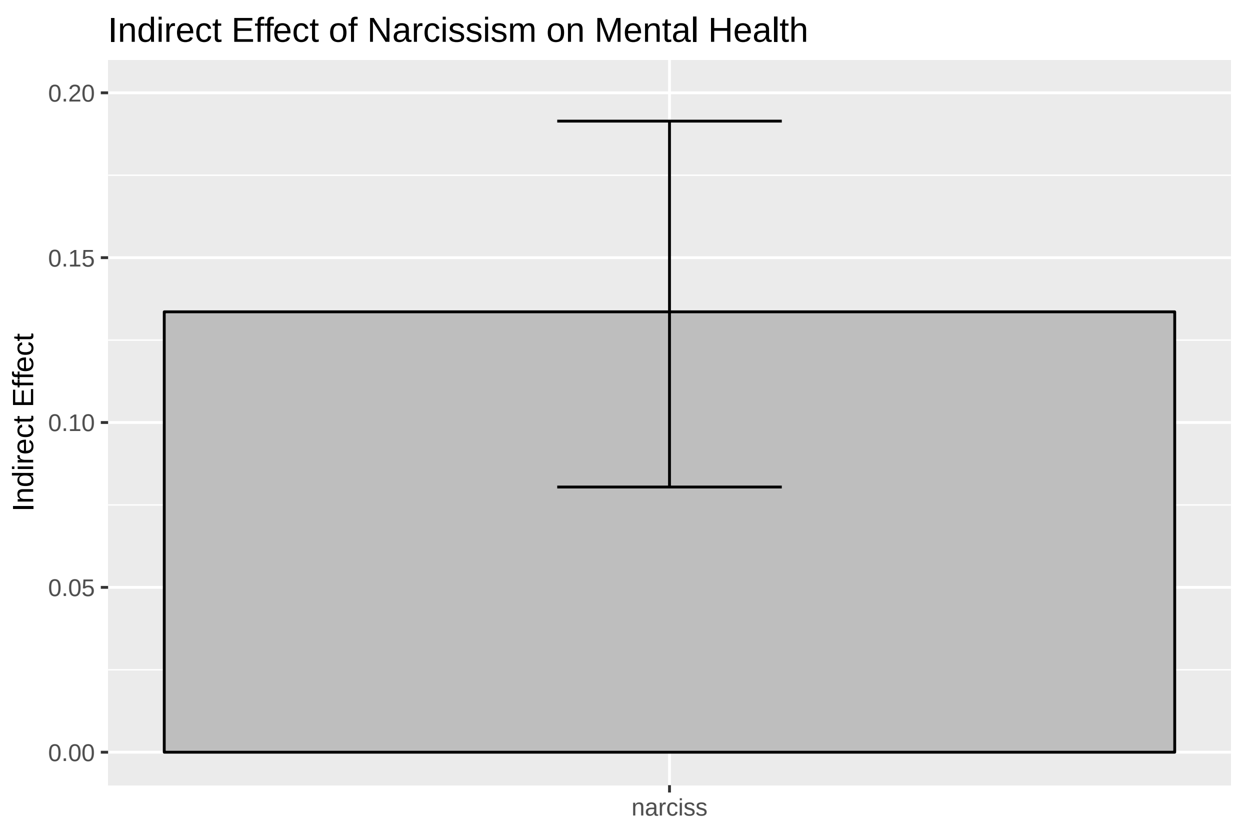
There are many options for plotting the indirect effect. Here I'll give two examples: the first is a simple bar graph, and the second shows the full range of bootstrapped samples.

First let's do a simple bar graph. In order to plot our data, we want to convert it into a `data.frame`:

```r
indirect.effect.data <- data.frame(predictor = "narciss",
                                   indirect.effect = original.indirect.effect,
                                   lower.ci = confidence.intervals[1],
                                   upper.ci = confidence.intervals[2])
```

Next, I'll make a simple bar graph. I add a couple of additional details, making sure that the 0 point is included in the y-axis so that significance is clear.

```r
p1 <- ggplot(indirect.effect.data, aes(x = predictor, y = indirect.effect)) +
  geom_bar(fill = "grey", color = "black", stat = "identity") +
  geom_errorbar(aes(ymin = lower.ci, ymax = upper.ci), width = 0.2) +
  coord_cartesian(ylim = c(0, 0.2)) + # makes sure the y-axis goes through zero
  xlab("") +
  ylab("Indirect Effect") +
  ggtitle("Indirect Effect of Narcissism on Mental Health")
p1
```

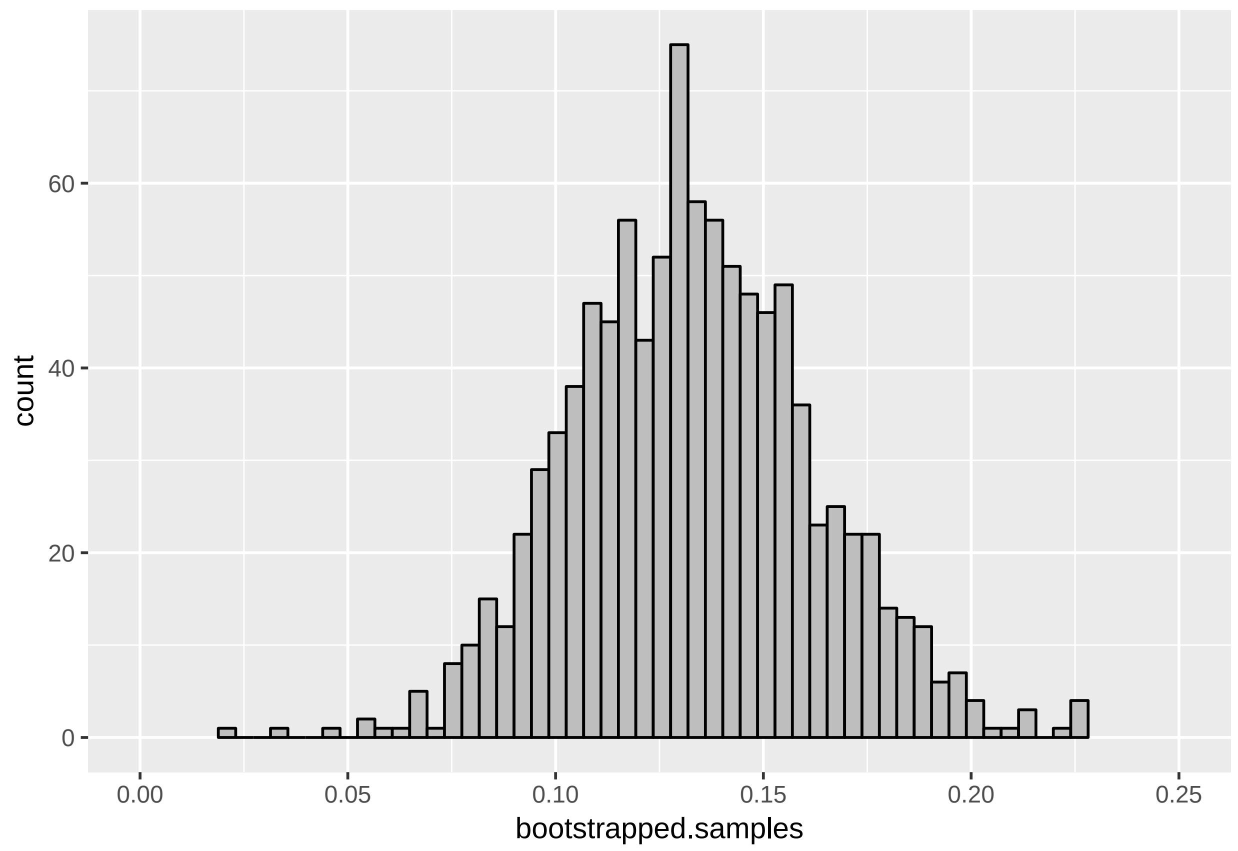


Now let's show the full range of bootstrapped samples! First, we'll keep our data frame from above to mark the relevant confidence intervals and data. But we'll also make another data frame using all of our bootstrapped samples:

```r
bootstraps <- as.data.frame(bootstrapped.samples)
```

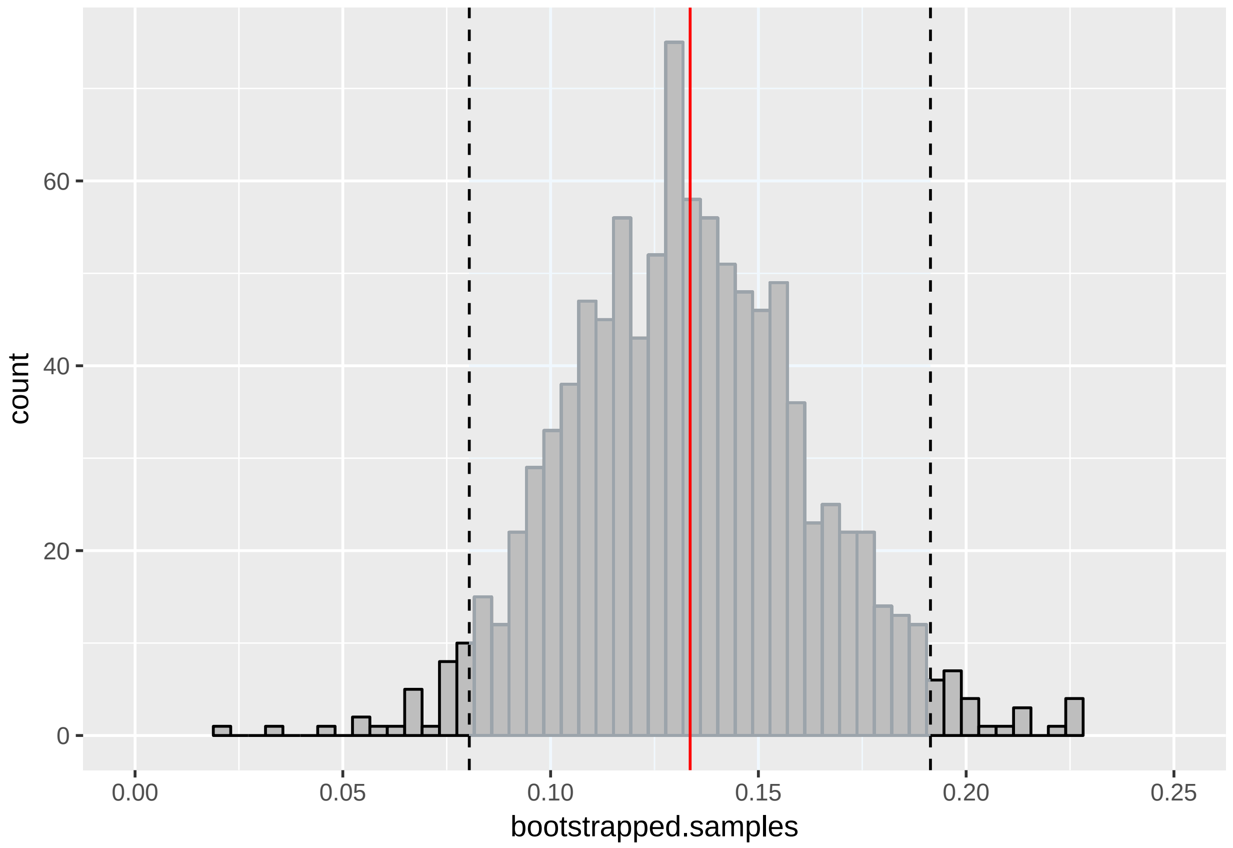
Now we'll incrementally build up a plot, starting by showing all of the samples of the data in a histogram, making sure to inclue the zero point on the x-axis:

```r
p2 <- ggplot(bootstraps, aes(x = bootstrapped.samples)) +
  geom_histogram(bins = 50, color = "black", fill = "grey") +
  coord_cartesian(xlim = c(0, 0.25))
p2
```



Now we'll add in lines and shading to show where the empirical indirect effect and confidence intervals are:

```r
p2 <- p2 +
  geom_rect(aes(xmin = indirect.effect.data$lower.ci[1], xmax = indirect.effect.data$upper.ci[1],
              ymin = -Inf, ymax = Inf), alpha = 0.01, fill = "aliceblue") +
  geom_vline(xintercept = indirect.effect.data$indirect.effect[1], color = "red") +
  geom_vline(xintercept = indirect.effect.data$lower.ci[1], linetype = "dashed") +
  geom_vline(xintercept = indirect.effect.data$upper.ci[1], linetype = "dashed")
p2
```



We get more information about the sampling distribution than we did in the simple bar graph case: in fact, we can see that not a single one of our bootstrapped sample estimates of the indirect effect was less than or equal to zero! Showing either graph in homeworks and the final will be fine, it just depends on what kind of information you want to convey.

Reporting Mediation Effects via Path Diagrams

In addition to plotting the indirect effect, you'll also want to show the full story in terms of the mediation path diagrams we saw at the beginning of class. To this end, just report the effects you found on each of the arrows in the diagram, like the one displayed on the next page.

Note that this diagram does not show the indirect effect, so make sure to plot it in some way and report it as well!

4/21/2017 Update: Multiple Mediators, Multiple Predictors

The final may or may not have a question about mediation with multiple predictors or multiple mediators. There will be several ways in which you could answer this question, but I want you to be able to do all of the possibilities! To that end, I think the best option is for y'all to have a more full understanding of the mediation models so that you can manually make your own specialized indirect effect function you can then bootstrap over.

Recall that the indirect effect is $a * b$ in our path diagram, where:

- a = coefficient from model predicting the mediator(s) from the predictor(s)

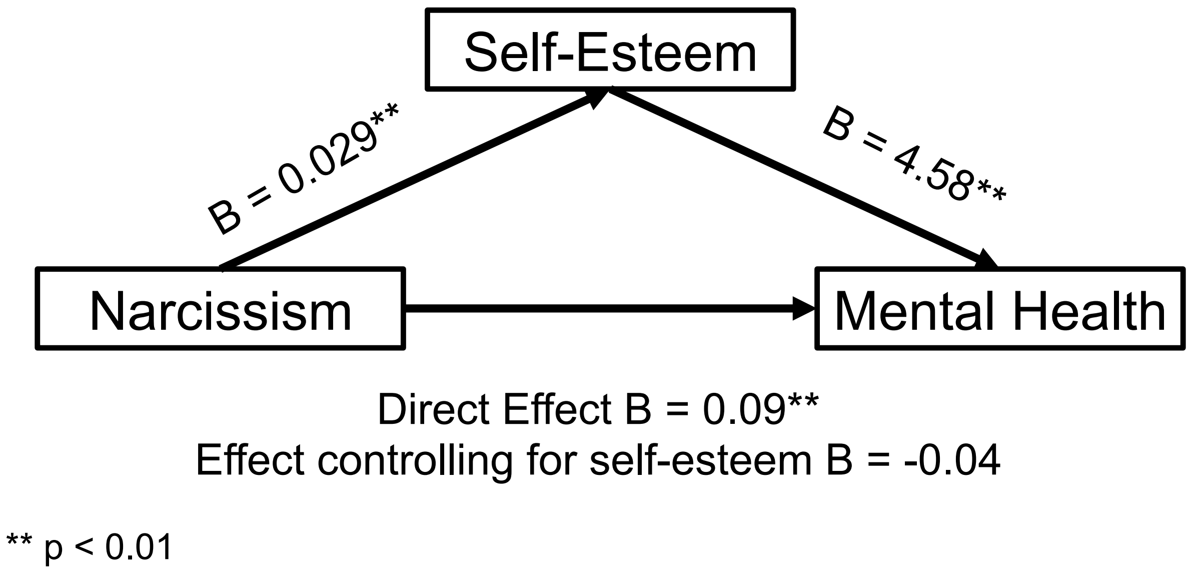


Figure 2: Reporting mediation effects with path diagrams.

- b = coefficient from model predicting DV from the mediator(s), *controlling for the predictor(s)*

The function I wrote for you above only fits models with one mediator and one predictor. What you can do, however, is write your own function on the fly that takes advantage of the fact that R allows global variables to appear in functions.

Let's use our same example dataset from above and add a couple of more variables (just randomly generated here):

```
d$predictor2 <- rnorm(nrow(d), mean = 3, sd = 2)
d$mediator2 <- rnorm(nrow(d), mean = 2, sd = 1)
```

Recall our old variables:

```
head(d)
```

```
##         se narciss    mental predictor2 mediator2
## 1 2.900000      19 -2.396314  2.7911795 2.6474571
## 2 3.800000      17  4.481908  2.7860191 2.8244036
## 3 4.000000      21  2.646707  2.9604385 0.6921227
## 4 2.444444       9 -1.960831  1.0617147 0.9825186
## 5 4.000000      24  1.922139  0.1203914 1.6783477
## 6 3.800000      23  3.701790  3.7463531 0.5725017
```

Our original predictor of interest was `narciss` and mediator was `se`.

Now, let's say that we want to test all possible mediations. There are 4 possibilities:

- Indirect effect of `narciss` on `mental` mediated by `se`
- Indirect effect of `predictor2` on `mental` mediated by `se`

- Indirect effect of `narciss` on `mental` mediated by `mediator2`
- Indirect effect of `predictor2` on `mental` mediated by `mediator2`

We have several options for how we can test these:

- Test each indirect effect separately. That is, our "a" and "b" models will only contain the mediator and predictor of interest.
- Control for other possible mediators, but not other possible predictors, for each indirect effect.
- Control for other possible predictors, but not other possible mediators, for each indirect effect.
- Control for other possible predictors AND other possible mediators in our "a" and "b" models.

The last option is the one with the lowest false positive rate, so we should probably go with that strategy. Now, what will we need to do?

- We'll have two "a" models: one predicting `se` from the two predictors, controlling for `mediator2`; and one predicting `mediator2` from the two predictors, controlling for `se`
- We'll only need on "b" model: predicting `mental` from both predictors and both mediators

This will give us each unique combination of the "a * b" indirect effect for the four possible mediation scenarios we laid out above.

Now I'll show you how to write your own custom function *within* the replicate function we've used above. Recall that before, I had given you the `compute.indirect.effect()` function, which can take in any `data.frame` and column strings you specified. What we can do instead, however, is write a function of our own that doesn't require being general enough to do that but is specific for our needs.

```r
compute.indirect.effect.global <- function(r = TRUE) {
  # sample the data
  s <- sample(nrow(d), replace = r)
  d.sampled <- d[s, ]

  # fit the a and b models
  a1.model <- lm(se ~ narciss + predictor2 + mediator2, d.sampled)
  a2.model <- lm(mediator2 ~ narciss + predictor2 + se, d.sampled)
  b.model <- lm(mental ~ narciss + predictor2 + se + mediator2, d.sampled)

  # extract the "a" coefficients for each predictor-mediator pair
  a.se.narciss <- summary(a1.model)$coefficients["narciss", "Estimate"]
  a.se.predictor2 <- summary(a1.model)$coefficients["predictor2", "Estimate"]
  a.mediator2.narciss <- summary(a2.model)$coefficients["narciss", "Estimate"]
  a.mediator2.predictor2 <- summary(a2.model)$coefficients["predictor2", "Estimate"]

  # extract the "b" coefficients for each mediator
  b.se <- summary(b.model)$coefficients["se", "Estimate"]
  b.mediator2 <- summary(b.model)$coefficients["mediator2", "Estimate"]

  # compute the indirect effects
  indirect.se.narciss <- a.se.narciss * b.se
  indirect.se.predictor2 <- a.se.predictor2 * b.se
  indirect.mediator2.narciss <- a.mediator2.narciss * b.mediator2
  indirect.mediator2.predictor2 <- a.mediator2.predictor2 * b.mediator2

  results <- data.frame(predictor = c("narciss", "narciss", "predictor2", "predictor2"),
                        mediator = c("se", "mediator2", "se", "mediator2"),
                        indirect.effect = c(indirect.se.narciss, indirect.mediator2.narciss,
                                            indirect.se.predictor2, indirect.mediator2.predictor2))
  return(results)
}

bootstrap2 <- replicate(1000, compute.indirect.effect.global(), simplify = F)
bootstrap2 <- do.call(rbind, bootstrap2)
```

Now that our output is a `data.frame` with all of the predictor-mediator pairs, we'll want to get confidence intervals for each.

```r
library(dplyr)
```

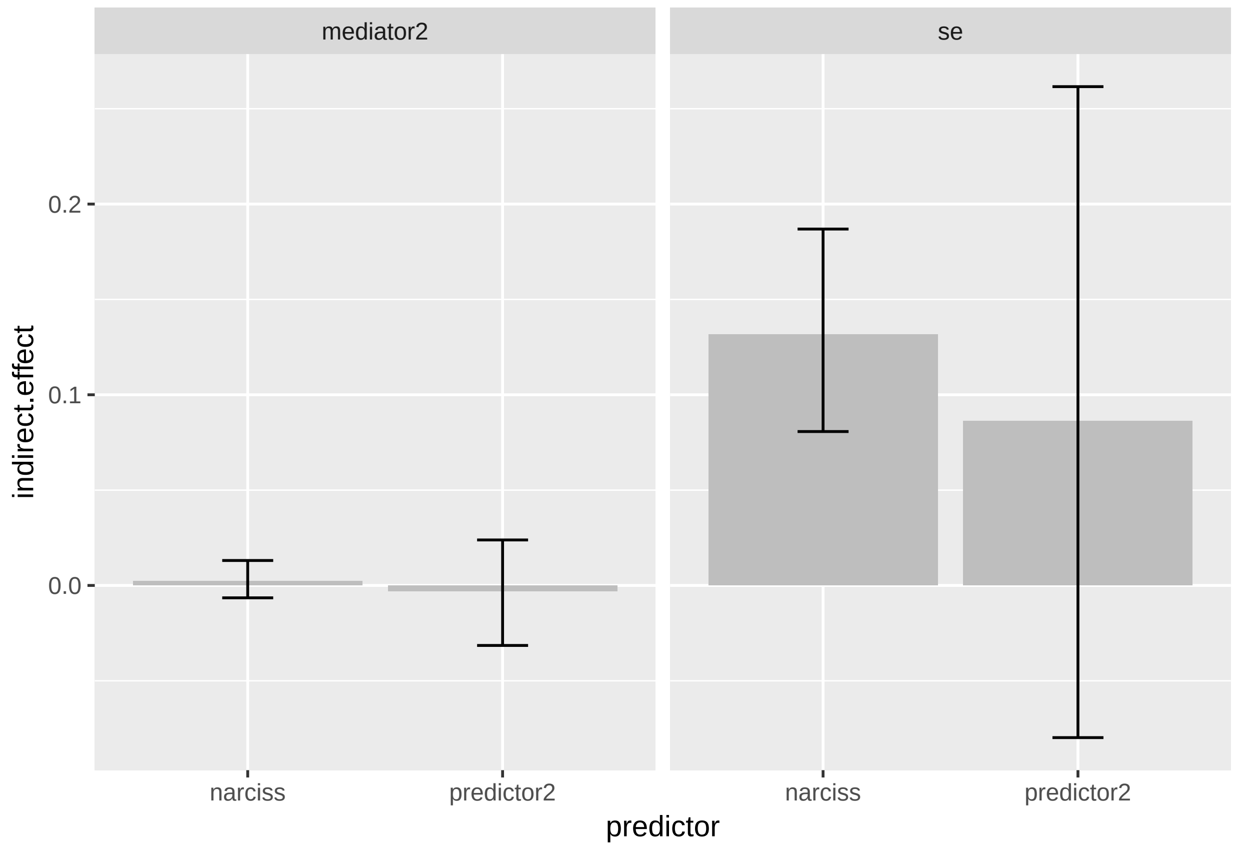
```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```r
bootstrap2.summary <- bootstrap2 %>%
  group_by(predictor, mediator) %>%
  summarise(lower.ci = quantile(indirect.effect, probs = 0.025),
            upper.ci = quantile(indirect.effect, probs = 0.975))
# compute original indirect effects
original.indirect.effects <- compute.indirect.effect.global(r = FALSE)
# merge with bootstrapped CIs
bootstrap2.summary <- merge(bootstrap2.summary, original.indirect.effects, by = c("predictor", "mediato

p.all.effects <- ggplot(bootstrap2.summary, aes(x = predictor, y = indirect.effect)) +
  geom_bar(fill = "grey", stat = "identity") +
  geom_errorbar(aes(ymin = lower.ci, ymax = upper.ci), width = 0.2) +
  facet_wrap(~ mediator)
p.all.effects
```


mediator2
se
indirect.effect
0.2
0.1
0.0
narciss
predictor2
narciss
predictor2
predictor