PSY653, Unit 10, In class activity – Advanced Mediation

Dataframe: obs.csv

In this activity you will build on your notebook named: BAC_Notebook_Mediation.Rmd. Last week, the underlying mediation model that we tested was: x = typ_drks, m = alcexp, y = alc_gm. That is, we examined the extent to which the effect of typical drinking pattern on alcohol consumed on the 21st birthday was mediated by alcohol expectancies for the role that alcohol would play in the birthday celebration (measured 1 week prior to the birthday). We found that alcexp explained (i.e., mediated) a part of the effect. Today we will expand our analysis to consider a potential moderator of this mediation model – partying mood. It is reasonable to speculate that the degree to which the woman was in a partying mood on the day of her birthday would affect the extent to which both typical drinking patterns and alcohol expectancies influenced actual drinking on the day of the event.

1. Draw a diagram to depict the model.

2. At the bottom of your notebook, create a first level header called: Test an advanced mediation model. Then create a second level header called: Evaluate which effects are moderated.

    a. Create a third level header called: Determine if pmood moderates the effects of typ_drks on alc_gm. Fit a linear model where alc_gm is regressed on typ_drks, pmood, and the interaction of the two. Note if the interaction term is significant.

    b. Create a third level header called: Determine if pmood moderates the effects of alcexp on alc_gm. Fit a linear model where alc_gm is regressed on alcexp, pmood, and the interaction of the two. Note if the interaction term is significant.

    c. From these model, you see that the b-path of the mediation model differs as a function of our moderator, therefore, there will not be one indirect effect that works for the entire population. Rather, we need to calculate the indirect effect at different levels of the moderator. We will choose 3 prototypical values of pmood – a relatively low value (4), a moderate/middle value (6), and a high value (8). (Note that we could do the same thing with any other set of values of pmood, including the mean and +/- 1 SD from the mean.)

3. First we need to create new versions of our moderator that are centered at these prototypical values of interest. Create a second level header called: Create centered versions of pmood. Use mutate to create three new versions of pmood (pmood_4, pmood_6, pmood_8) centered at 4, 6, and 8 respectively.

4. Create a second level header called Set of Model 1 Equations. Here you will estimate the total effect (c-path) of x (typ_drks) on y (alc_gm) at varying levels of pmood.

    a. Create a third level header called: Model 1 Lo. Regress alc_gm on typ_drks, pmood_4 and the interaction of the two.

    b. Create a third level header called: Model 1 Mid. Regress alc_gm on typ_drks, pmood_6 and the interaction of the two.

    c. Create a third level header called: Model 1 Hi. Regress alc_gm on typ_drks, pmood_8 and the interaction of the two.

    d. In these models, the effect for typ_drks is the c-path at each level of moderator (i.e., 4, 6, 8).

5. Create a second level header called: Model 2. Fit a linear regression model where alcexp is regressed on typ_drks. This is the a-path of the mediation model, notice that we have just one a-path because it is not moderated.

6. Create a second level header called Set of Model 3 Equations. Here you will estimate the direct effect of x (typ_drks) on y (alc_gm) at varying levels of pmood (i.e., c'-paths) and the effect of m (alcexp) on y (alc_gm) at varying levels of pmood (i.e., b-paths).

    a. Create a third level header called: Model 3 Lo. Regress alc_gm on typ_drks, alcexp, pmood_4 and both relevant interactions (typ_drks*pmood_4 and alcexp*pmood_4).

b. Create a third level header called: Model 3 Mid. Regress alc_gm on typ_drks, alcexp, pmood_6 and both relevant interactions (typ_drks*pmood_6 and alcexp*pmood_6).

c. Create a third level header called: Model 3 Hi. Regress alc_gm on typ_drks, alcexp, pmood_8 and both relevant interactions (typ_drks*pmood_8 and alcexp*pmood_8).

d. In these models, the effects of typ_drks and alcexp represent the simple slopes for the c' and b-paths at each corresponding level of the moderator.

7. Create a second level header called: Calculate the indirect effect. Pull out the estimates and standard errors for the a-path and the b-paths of the mediation model (note that you will have three b-paths) and then calculate the product term for the indirect effects (you will have three indirect effects as well).

8. Create a second level header called: Use RMediation to calculate the CI for the indirect effects. Use the medci function to construct a 95% CI for the indirect effect. Note that you will need to execute medci three times, once for each level of pmood.

9. Create a second level header called: Use boot package to bootstrap the CI for the three indirect effects. Use the techniques that you learned in Unit 10 to construct the CIs for the three indirect effects using bootstrap resamples.

10. Study the full set of results and write up a paragraph to describe the results.

Test an advanced mediation model
Evaluate which effects are moderated
Determine if pmood moderates the effects of typ_drks on alc_gm
```{r}

moder1 <- lm(alc_gm ~ typ_drks + pmood + typ_drks*pmood, data = obs)
ols_regress(moder1)

```

Determine if pmood moderates the effects of alcexp on alc_gm
```{r}

moder2 <- lm(alc_gm ~ alcexp + pmood + alcexp*pmood, data = obs)
ols_regress(moder2)

```

Create centered versions of pmood - choose values of 4, 6, and 8
```{r}

obs <- obs %>%
  mutate(pmood_4 = pmood - 4,
         pmood_6 = pmood - 6,
         pmood_8 = pmood - 8)

```

Explore Model 1
Model 1 Lo: regress y on x, z, and x*z - center z at low value
```{r}

model.xy_lo <- lm(alc_gm ~ typ_drks + pmood_4 + typ_drks*pmood_4, data = obs)
ols_regress(model.xy_lo)

```

Model 1 Mid: regress y on x, z, and x*z - center z at middle value
```{r}

model.xy_mid <- lm(alc_gm ~ typ_drks + pmood_6 + typ_drks*pmood_6, data = obs)
ols_regress(model.xy_mid)

```

Model 1 Hi: regress y on x, z, and x*z - center z at high value
```{r}

model.xy_hi <- lm(alc_gm ~ typ_drks + pmood_8 + typ_drks*pmood_8, data = obs)
ols_regress(model.xy_hi)

```

````
## Model 2
```{r}

model.xm <- lm(alcexp ~ typ_drks, data = obs)
ols_regress(model.xm)

```

## Explore Model 3
### Model 3 Lo: regress y on x, m, z, x*z, and m*z - center z at low value
```{r}

model.xmy_lo <- lm(alc_gm ~ typ_drks + alcexp + pmood_4 + typ_drks*pmood_4 + alcexp*pmood_4, data = obs)
ols_regress(model.xmy_lo)

```

### Model 3 Mid: regress y on x, z, and x*z - center z at middle value
```{r}

model.xmy_mid <- lm(alc_gm ~ typ_drks + alcexp + pmood_6 + typ_drks*pmood_6 + alcexp*pmood_6, data = obs)
ols_regress(model.xmy_mid)

```

### Model 3 Hi: regress y on x, z, and x*z - center z at high value
```{r}

model.xmy_hi <- lm(alc_gm ~ typ_drks + alcexp + pmood_8 + typ_drks*pmood_8 + alcexp*pmood_8, data = obs)
ols_regress(model.xmy_hi)

```

## Calculate the indirect effects at each level of pmood
```{r}

# Pull out coefficients and standard errors for a and all b paths
path.a <- coefficients(model.xm)["typ_drks"]
se.path.a<-sqrt(vcov(model.xm)["typ_drks","typ_drks"])

path.b_lo <- coefficients(model.xmy_lo)["alcexp"]
se.path.b_lo<-sqrt(vcov(model.xmy_lo)["alcexp","alcexp"])

path.b_mid <- coefficients(model.xmy_mid)["alcexp"]
se.path.b_mid<-sqrt(vcov(model.xmy_mid)["alcexp","alcexp"])

path.b_hi <- coefficients(model.xmy_hi)["alcexp"]
se.path.b_hi<-sqrt(vcov(model.xmy_hi)["alcexp","alcexp"])

path.a
path.b_lo
path.b_mid
path.b_hi

# Calculate indirect effects
ab_lo <- path.a*path.b_lo
ab_lo

ab_mid <- path.a*path.b_mid
ab_mid

ab_hi <- path.a*path.b_hi
ab_hi

```
````

Use RMediation to calculate CI for indirect effects
```{r}

medci(path.a, path.b_lo, se.path.a, se.path.b_lo, alpha = 0.05, plot=TRUE, plotCI=TRUE)
medci(path.a, path.b_mid, se.path.a, se.path.b_mid, alpha = 0.05, plot=TRUE, plotCI=TRUE)
medci(path.a, path.b_hi, se.path.a, se.path.b_hi, alpha = 0.05, plot=TRUE, plotCI=TRUE)

```

Use boot package to bootstrap confidence intervals
```{r}

boot.med <- function(data, indices){
  data <-data[indices,]

  model.xm <- lm(alcexp ~ typ_drks, data=data)
  model.xmy_lo <- lm(alc_gm ~ typ_drks + alcexp + pmood_4 + typ_drks*pmood_4 + alcexp*pmood_4, data = data)
  model.xmy_mid <- lm(alc_gm ~ typ_drks + alcexp + pmood_6 + typ_drks*pmood_6 + alcexp*pmood_6, data = data)
  model.xmy_hi <- lm(alc_gm ~ typ_drks + alcexp + pmood_8 + typ_drks*pmood_8 + alcexp*pmood_8, data = data)

  path.a <- coefficients(model.xm)["typ_drks"]

  path.b_lo <- coefficients(model.xmy_lo)["alcexp"]
  path.b_mid <- coefficients(model.xmy_mid)["alcexp"]
  path.b_hi <- coefficients(model.xmy_hi)["alcexp"]

  ab_lo <- path.a*path.b_lo
  ab_mid <- path.a*path.b_mid
  ab_hi <- path.a*path.b_hi

  return(c(ab_lo, ab_mid, ab_hi))
}

medboot<-boot(data=obs, statistic = boot.med, R = 15000)
medconfint1 <- boot.ci(medboot, index=1, conf=(.95), type=c("bca"))
medconfint2 <- boot.ci(medboot, index=2, conf=(.95), type=c("bca"))
medconfint3 <- boot.ci(medboot, index=3, conf=(.95), type=c("bca"))
print(medconfint1)
print(medconfint2)
print(medconfint3)

```