Mediation Analysis: Expanding From One Mediator to Multiple Mediators

Brittanie Boone

Statistics Department, Carnegie Mellon University

Project Advisor: Howard Seltman

May, 2012

Abstract

Mediation analysis is the process of determining whether or not variables acting as an in-between step, called mediators, are present when looking at the relationship between an independent variable X and a dependent variable Y. An educational study was proposed to analyze the effect of different classroom environments on students performance. This study also measured if the student is on-task or off-task, and if off-task what type of distraction is the student distracted by. It is believed that the particular types of distractions are the mediators and act as an in-between step between the type of classroom environment and how well the students perform. This is what motivated the idea of multiple mediators and the question of how to deal with them. However, previous research that has been done has only looked at single mediators. My research is based on the Product of Coefficients approach (Sobel, 1982). Multiple mediators were analyzed separately or used principal component analysis. Simulated data was used to determine how well these tactics accurately detect mediation. It appears that detecting mediation when there are multiple mediators present is very feasible, and this paper will show the results of this analysis. Finally, this paper also compiled the functions into a statistical package in R, making it easier to analyze data with multiple mediators.

1 Introduction

There has been limited exploration of different methods of Mediation analysis. General cases of one mediator and one outcome seem to be the more focused research area of mediation analysis. This paper will focus on implementation and evaluation of different methods of mediation analysis in R. We will look into mediation analysis with multivariate dependent variables as well as multiple mediators. Currently there is limited software that performs mediation analysis. This paper will also present a package in R that was created to perform multivariate mediation analysis and corresponding power analysis based on simulated data. This package is used for exploring how well mediation analysis performs given whether mediation is present or not.

The initial idea of this exploration comes from an educational research proposal that is looking at how different classroom environments may affect a student's level of paying attention. Students are initially assigned to different type of classrooms; each classroom is different based on the amount of relevant and irrelevant posters and students school work hung on the walls. Then each student is watched to determining the time spent off-task and what the student was distracted by. There are 6 different categories of distraction, and for each distraction type the fraction of time spent off task was recorded. One analysis that is proposed is determining whether a mediated effect is present for the time spent off task and a student's outcome. It may be possible that the different classrooms affect the types and times of distraction and have no direct effect on a student's score. It may be possible that only certain types of distractions are actual mediators. Thus, when analyzing possible mediated effects it is important to figure out a way to use mediation analysis that can accurately predict significant mediation if it is present.

There are many different possibilities for how multiple mediators may affect a student's outcome. There may be some types of distraction that have no effect on how well a student is performing academically and others with stronger effects. There may also be two types of distractions that have an interaction. This analysis will focus on how well the mediation analysis works in these types of situations. When evaluating the mediated effect with multiple mediators, analyses will focus on ways to minimize the number of mediated effects without compromising the accuracy of detection by looking into principle component analysis, factor analysis, and other dimension reduction techniques. When thinking about the

problem of multiple mediators, the problem of multiple dependent variables also emerged. It is possible that dimension reduction techniques can be used to resolve this issue as well.

Currently there is a package in R called mediation that runs mediation analysis that consists of one mediator and one dependent variable. It requires the user to input the equation that predicts the mediator and the equation that predicts the outcome, as well as to specify which variable is the mediator and which is the outcome. The summary output gives estimates for the average causal mediation effect, direct effect, total effect, and proportion of total effect mediated. It also performs sensitivity analysis to see how well the ignorability assumption holds. Given the mediated model and the outcome model, the sequential ignorability assumption states that the correlation between the error terms for each equation is 0. This correlation is denoted as ρ, and sensitivity analysis provides estimates of ρ.

Advantages of this package are that the package allows for linear and non-linear models, the outcome variable can be continuous or binary, and the mediator can be continuous, ordered, or binary. However, the package in R does not actually give a p-value in the results. It also cannot be used for multiple mediators or multiple dependent variables. This new mediation package will explore and address these issues.

2 Mediation

2.1 Mediation Analysis

Mediation analysis is needed to determine whether or not mediators are present when looking at the relationship between two random variables X and Y; mediators are variables that act as an in-between step when looking at the effect of X on Y, where X causes a mediator M, and M is actually the cause of Y. Related variable types are confounding, covariate, moderator or interaction effect. Confounding variables would be when a third variable can actually explain both X and Y. A covariate is a variable that may relate to X and/or Y but does not significantly impact the relationship between X and Y. Finally, a moderator or interaction variable Z, would be one that may alter the relationship between X and Y at different levels of Z. The three main regression equations used for mediation analysis on a single-mediator model are as

follows:

$$Y = i_1 + cX + e_1 \tag{1}$$

$$Y = i_2 + c'X + bM + e_2 \tag{2}$$

$$M = i_3 + aX + e_3 \tag{3}$$

The total effect of mediation analysis is given by:

$$\text{Total Effect} = a * b + c'$$

The direct effect of mediation analysis is given by:

$$\text{Direct Effect} = c'$$

The mediated effect is given by:

$$\text{Mediated Effect} = c - c' \ = \ a * b$$

There are three main approaches to statistical mediation analysis:

1. Causal Steps
2. Difference in Coefficients
3. Product of Coefficients

Causal steps is the most common approach. If mediation is present then the following would need to be met: the X coefficient would be significant in (1), the X coefficient would be significant in (3), M would be significant in (2) and the X coefficient would decrease. This would mean partial mediator was present; if the coefficient became non-significant then this would be an example of complete mediation.

The Difference in Coefficients method only uses equations (1) and (2) and takes the difference $c - c'$ to represent the mediated effect. $c - c'$ refers to the reduction in the independent variable effect on the

dependent variable adjusted for the mediator. The difference estimate is divided by the standard error of the difference in order to obtain a test statistic. This method was suggested by Judd and Kenny (1981).

The other method of mediation analysis is what this paper will focus on, and is called the Product of Coefficients. This method only uses $a * b$ from equations (2) and (3) and refers to the fact that mediation depends on the extent to "X" changes the mediator, a, and the extent to which the mediator affects the outcome variable, b. The significance test is same as in the Difference in Coefficients except you divide by the standard error of the product. The Sobel test (Sobel 1982) has shown the standard error is approximately equal to:

$$SE_{ab} = \sqrt{(a^2 * \text{se}_b^2) + (b^2 * \text{se}_a^2)}$$

The test statistic is derived by:

$$Z_{ab} = \frac{\text{Mediated Effect}}{\text{SE}_{ab}} = \frac{a * b}{\text{SE}_{ab}}$$

The Sobel test, however, is very conservative. Firstly it is based on the assumption that a and b are independent. It also uses a normal approximation which presumes a symmetric distribution, but falsely presumes symmetry. Thus, the distribution of $a * b$ is highly skewed away from zero.towards the direction of $a * b$. This method will be used for analysis in this paper. We will also use a bootstrap method to allow dependency and skewness.

The following are the assumptions of mediation analysis:

- No misspecification of causal order (i.e. casual order is $X \rightarrow M \rightarrow Y$)
- No misspecification of causal direction
- No misspecification due to unmeasured variables that cause variables in the mediation analysis
- No misspecification due to imperfect measurement of X or M
- Ignorability assumption (i.e. the residuals of equations (2) and (3) are independent)
- There is no interaction between X and M in their effects on Y

3 Simulations with One Mediator and Analysis

This section describes the simulations functions written as part of this research that deals with mediation analysis with one mediator. The first step in simulating mediation analysis in R is to first simulate variables X, M, and Y. For mediation to work we want X to predict M and only M to predict Y. Thus, the distributions are given by:

1. X $\sim$ Normal(x.mean,x.sd)

2. M $\sim$ Normal(mean=Bx*x,sd=m.sd)

3. Y $\sim$ Normal(mean=Bm*m, sd=y.sd)

The simulation function, "mediate.sim", allows Bx, Bm, sd, and the number of observations to be chosen before running it. When running this function it is also important to make sure the method="normal". The default standard deviation for all variables is 1. Then the function returns a data frame containing X, M, and Y. The next step is to analyze the data and return a p-value of the mediated test statistic. The analyze function,"mediate.analyze", takes in the three vectors, finds the mediated effect ab, calculates the se, and then produces a p-value corresponding to the Z test of a normal distribution. Then the sapply function in R can be used to generate multiple p-values to calculate the power. A gamma error and an alternate bootstrap approach will also be used for comparison.

To see how well the Sobel test works, the power was found given there is no mediation. Setting Bx=0, Bm=1, and n=100, if there was mediation present then the null would be rejected 5% of time. After running 100 simulations, the null was rejected on average 5% of the time. (Results varied for each simulation.) Since we got around a 0.05% rejection rate, as Bx is increased this rate should increase as well until it gets to 100%, which corresponds to correctly rejecting the null when the null is false 100% of the time. Using small intervals of 0.05 for Bx and running it for multiple values of Bm the power curves are below:

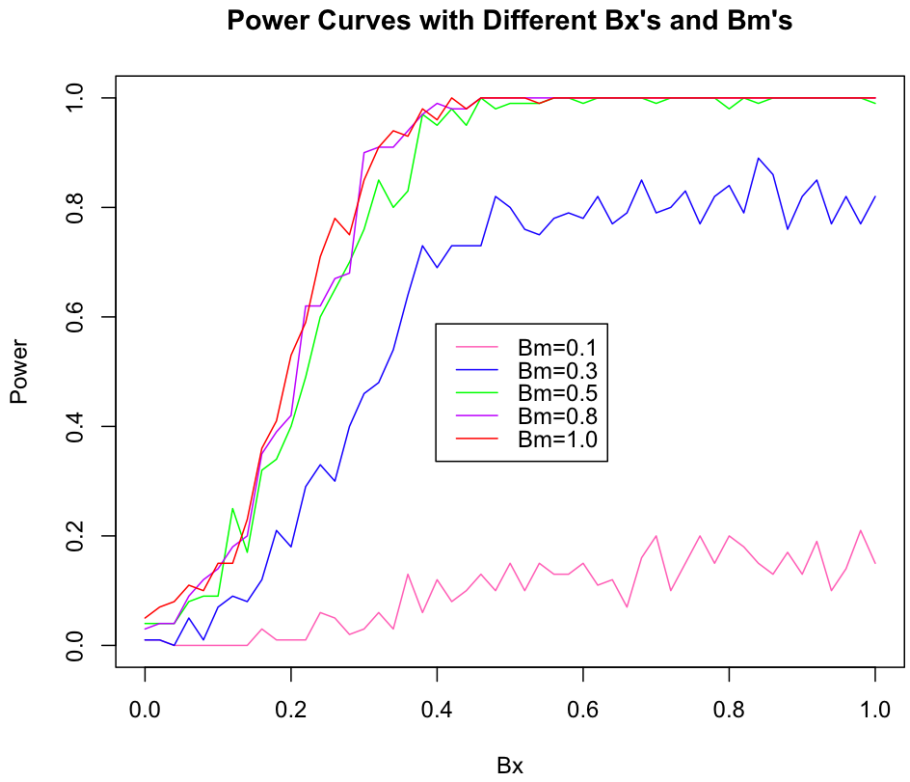

Figure 1: Power Curves

We can see from Figure 1 the bottom curve has the lowest power, as expected. Increasing Bm by 0.2 increases power dramatically. Once Bm=0.5 the results are as similar to Bm=0.8 and Bm=1.0. They all seem to converge to 100% power around Bx=0.4.

For comparison, I also created a bootstrap function that resampled the initial simulation data to calculate a new standard error. The number of bootstrap iterations used for calculating the standard errors can be selected when calling the function, but the default is 200. After the 200 new standard errors are found, the standard deviation of these standard errors are used as our new standard error in the analyze function. After running the bootstrap simulation 500 times for time purposes, the null was rejected 8% of the time.

Next, to test for the robustness of the Sobel test, some non-normal error was looked it. Gamma error was used and then the same analyze function. After trying multiple alpha and beta levels here are the results of the gamma errors with different alpha and betas compared to the normal error:

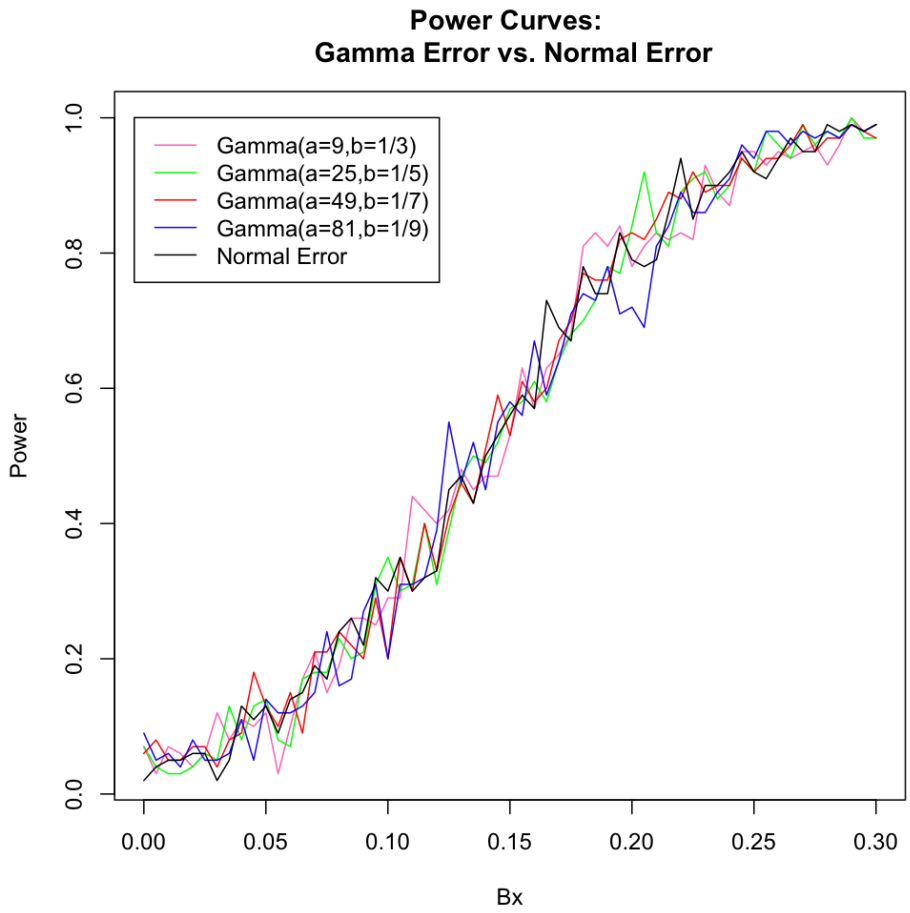

Figure 2: Power Curves with Gamma Error

Since all of the curves are fairly similar we can conclude that the sobel test is robust for large values of n. These curves all had n=200 and the simulations were run 100 times. These results make sense given the Central Limit Theorem states the mean of a sufficiently large independent variables will be approximately normally distributed. In order to look at the effect of sample sizes we ran multiple simulations with different sample sizes.

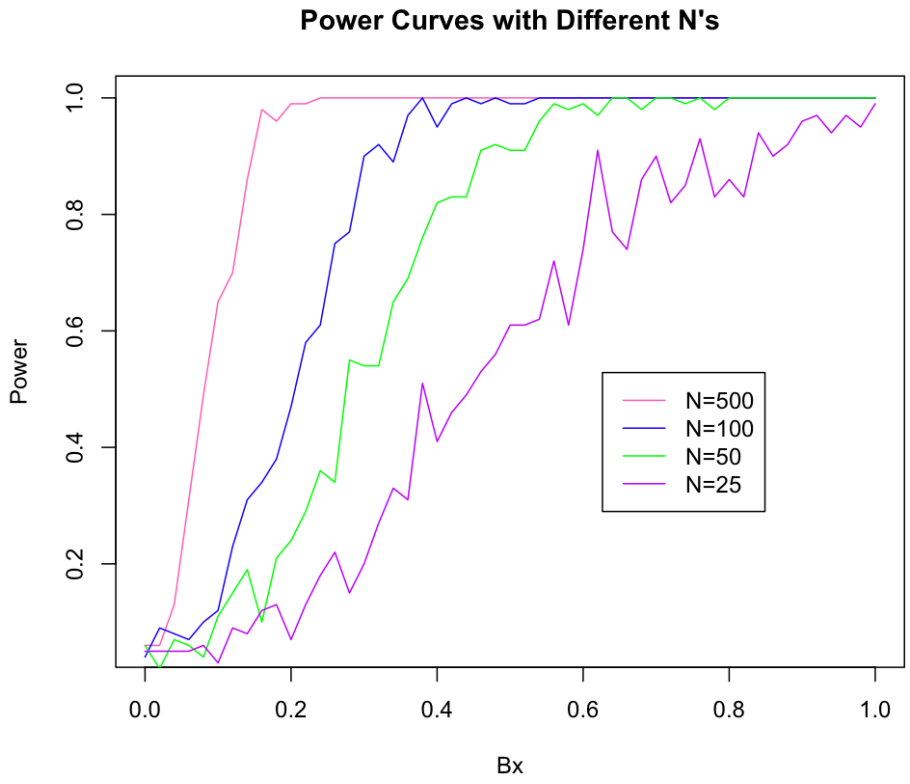

Figure 3: Power Curve with Different N's

Figure 3 shows the power curve for sample size of 500, 100, 50, and 25. There is a significant decrease in power as n decreases. The Bx hat converges to 100% power is increased too, meaning for smaller samples there needs to be a larger value for Bx in order to obtain a significant result.

The following plots will look at how the power changes as Bx and Bm changes. Figure 4 shows how power changes as Bx is increased at a rate of 0.01 for sample sizes n=30, n=100, and n=200, and each curve on the plot corresponds to a particular level of Bm from 0 to 1 and goes in increments of 0.01. Figure 5 shows how power changes as Bm is increased at a rate of 0.01 for sample sizes n=30,n=100, and n=200, and each curve on the plot corresponds to a particular level of Bx from 0 to 1 and goes in increments of 0.01. These plots were included to give a better picture of how much power can be estimated given a certain Bx or Bm.

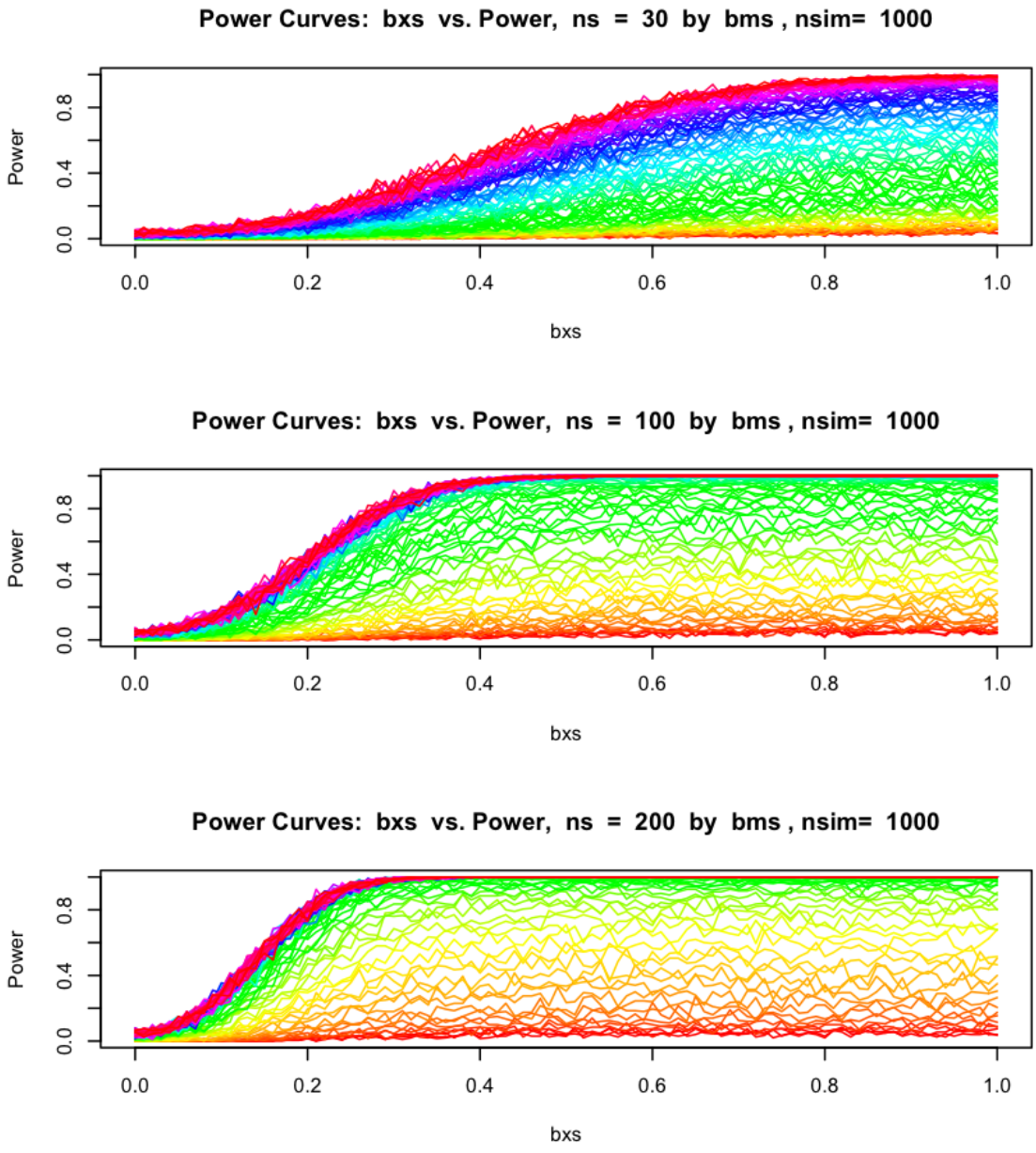

Figure 4: Power Curve: Bx vs. Power by Bm from 0 to 1, n=30, 100, 200

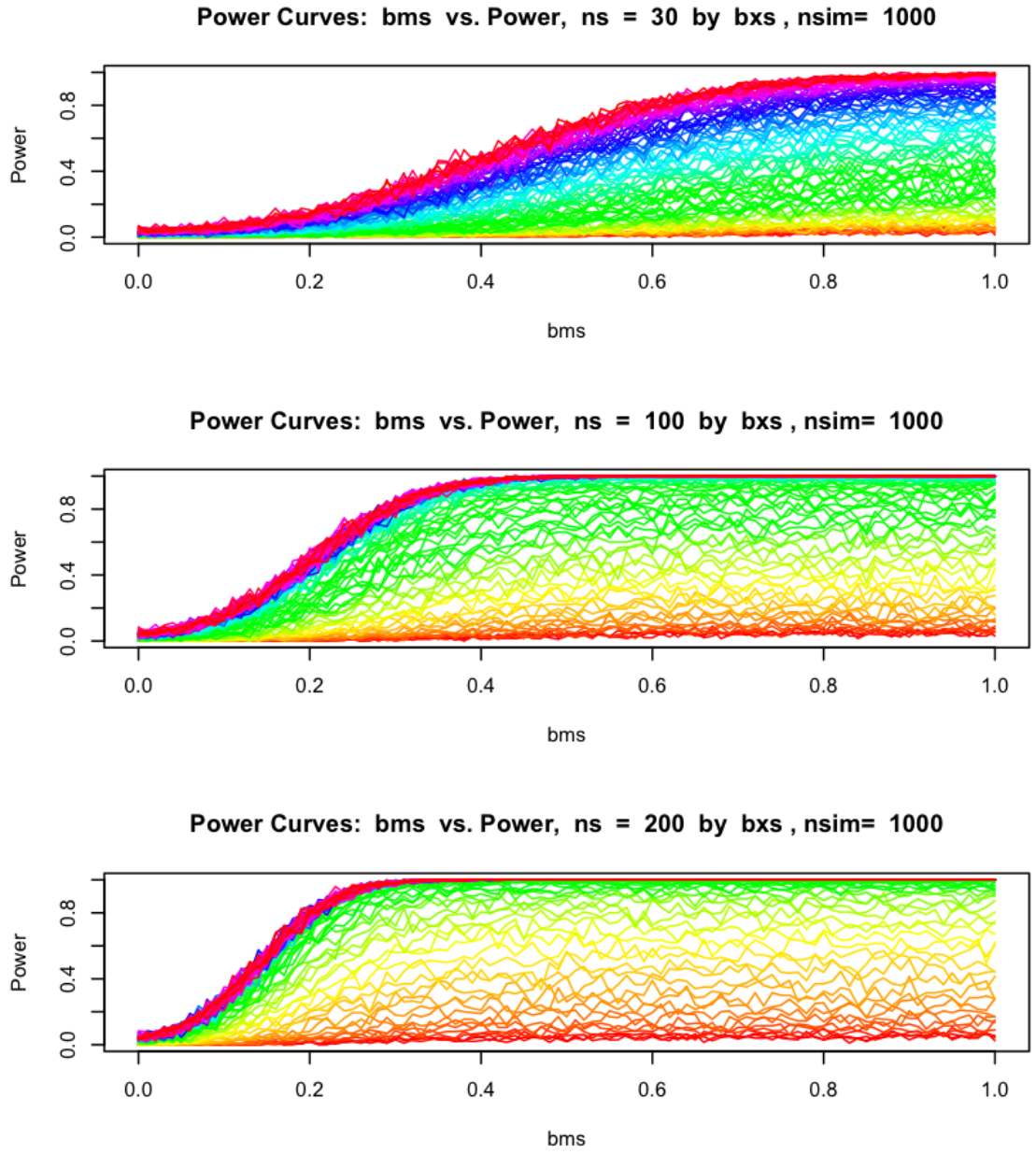

Figure 5: Power Curve: Bm vs Power by Bx from 0 to 1, n =30, 100, 200

An important take away of Figure 4 and Figure 5 is that Bx and Bm have a fairly equal weight on the affect on power. In more simpler terms, if Bx is increased by 0.1 then it would have the same affect on power as increasing Bm by 0.1. This makes sense because mediation needs X to predict M and M to predict Y. When the Bx is increased this means that X predicts M stronger than the smaller Bx. When Bm is increased this means that M predicts Y stronger than the smaller Bm. Both of which are equally as important when calculated the mediated effect and whether or not is significant.

4 Multiple Mediation Analysis

After running all of these different functions and creating the different power curves, it now seems feasible to expand the mediation analysis of just one dependent variable to mediation analysis with multiple mediators as well as multiple dependent variables that are correlated with each other.

4.1 Simulations with Multiple Mediators

In the introduction the possibility of multiple mediators was discussed. This section will empirically show the different methods of handling multiple mediators and determine which is the best method to use. There are multiple methods in which to handle multiple mediators. One method is to treat the mediated effects as independent and analyze each effect separately. Another method is to use principal component analysis to reduce the dimensions of m and then analyze the reduced predictions separately. Finally, there could also be a combination of both, where some mediators are correlated with each other and other mediators are uncorrelated with the rest of the mediators.

Simulating these variables in R will be more complicated than before. The following show the distributions:

1. X $\sim$ Normal(x.mean,x.sd)
2. for each M_i for $i = 1, ..., k$ where k is the number of mediators: $M_i \sim$ Normal(mean=Bx_i*x,sd=$m.sd_i$)
3. Y $\sim$ Normal(mean=$\sum(Bm_i * M_i)$, sd=y.sd)

The first method to analyze multiple mediators is to analyze each mediator separately.This method can be specified in the "mediate.analyze" and "sim.analyze functions" as method="multivariate.m.sep". Therefore each m will have it's own $a*b$ that will be used to find the test statistic and p-value. This method is the simplest to think about, but when there are a lot of mediators might be very complicated to analyze. The equations (2) and (3) from before now look like:

$$Y = i_2 + c'X + \sum_{i=1}^{\text{num.m}} b_i M_i + e_2 \tag{4}$$

$$M_i = i_3 + \sum_{i=1}^{\text{num.m}} a_i X + e_3 \tag{5}$$

The next method to analyze multiple mediators is to analyze the mediators using principal component analysis. This method can be specified in the "mediate.analyze" and "sim.analyze" functions as method="multivariate.m.pca". This method takes in all of the mediators and runs principal component analysis. PCA is a method of dimension reduction using an orthogonal transformation. The goal is to reduce the data by allowing the first principal component to have the largest variance so that it accounts for as much of the variability in the data as possible. Each additional principal component adds more variance, and if the number of principal components is equal to the number of mediators then it will account for 100 percent of the variance, as there would be no dimension reduction. The function allows one to specify the needed proportion of variance when running principal component analysis. The default is 0.7. Therefore if the first principal component prediction accounts for 70 or more percent of the variance then we will use the first principal component prediction as our combined M and only find one p-value that represents all of the mediators. If the first PC does not have at least 70 percent or the specified proportion of variance the function will use additional PC predictions until the needed proportion of variance is met. These equations are similar to (4) and (5); however, the number of terms depend on the number of principal components need to meet the minimum proportion of variance, needed.pc:

$$Y = i_2 + c'X + \sum_{i=1}^{\text{needed.pc}} b_i \text{Predicted.PC}_i + e_2 \tag{6}$$

$$\text{Predicted.PC}_i = i_3 + \sum_{i=1}^{\text{needed.pc}} a_i X + e_3 \tag{7}$$

The final method combines both of these ideas to analyze the mediated effects. This method can be specified in the "mediate.analyze" and "sim.analyze" functions as method="multivariate.m" This method will look at the data frame and determines which mediators will be analyzed separately and which will be analyzed using principal component analysis. Firstly a correlation matrix is computed, and given a certain minimum correlation, the function makes sure every mediator is at least this correlated with another mediator. If any of the mediators are not correlated with all of the others than it will be analyzed separately in the function and will be given it's own p-value. The necessary correlation can be specified to determine at what level should a mediator be considered not correlated with the rest of the mediators; the default is 0.20. Next, the mediators that are correlated with each other will go through principal component analysis. It will also be possible to specify the needed proportion of variance for the pca for this function as well, and the default for this is 0.70. The equations for determining a and b are a combination of (4) and (5), and (6) and (7):

$$Y = i_2 + c'X + \sum_{i=1}^{\text{num.m.sep}} b_i M_i + \sum_{i=\text{num.m.sep}+1}^{(\text{num.m.sep}+1)+\text{needed.pc}} b_i \text{Predicted.PC}_i + e_2 \tag{8}$$

$$M_i = i_3 + \sum_{i=1}^{\text{num.m.sep}} a_i X + \sum_{i=\text{num.m.sep}+1}^{(\text{num.m.sep})+1+\text{needed.pc}} a_i X + e_3 \tag{9}$$

The analysis will start by looking at data with two mediators. The first analysis was ran having both M_1 and M_2 come from mean=X*1, and Y coming from mean=M.1*1+M.2*1. It was ran with n=200 and then simulated 1000 times.

Using the first method that evaluates all mediators separately, the null hypotheses for mediator 1 and mediator 2 were both rejected 100 percent of the time. This means that there is significant evidence that mediation is present for both mediators correctly 100 percent of the time. Thus out of 1000 simulations, the null was correctly rejected always, which means there is 100 percent power.

When using the PCA method it reduced both mediators to one dimension 968 out of the 1000 times. The null that there is no mediation was rejected 100 percent of the time. In the 32 times that the method did not reduce the two mediators the null hypothesis for the first PC was rejected all 32 times, and the null hypothesis for the second PC was retained all 32 times.

Finally, the combined method was used. Out of the 1000, 966 times the M's met the minimum proportion of variance of 0.70. In all 966 simulations, the p-value that comes from the first principal component prediction was< 0.05. This means we would reject that there is no mediation for mediators 1 and 2 100 percent of the time. In the other 34 simulations the p-value from first principal component prediction was $< .05$ for all. However, in all 34 simulations the p-values that comes from the second principal component direct were all > 0.05. In other words, for these 34 simulations we would reject that there is no mediation for one mediator, but would retain the null that there is no mediation for the other mediator. It appears that we are making a type II error for the one of the mediators 3.4 percent of the time. This gives a power of 96.6 percent, which is very good for a simulation with a moderate effect size and a moderate sample size.

The next analyses will keep everything the same, except we will change bx's coefficients. The tables below display the results of the different methods for certain bx coefficients.

Method=Separate Mediation (n=200, Simulations=1000, bm=c(1,1), σ=1)		
	Power	
Bx	**Mediator 1**	**Mediator 2**
(1,1)	1.00	1.00
(1,0.5)	1.00	1.00
(1,0.2)	1.00	0.784
(1,0.1)	1.00	0.273
(0.5,0.1)	1.00	0.306
(.05,0.5)	1.00	1.0.0
(0.2,0.2)	0.787	0.793
(0.1,0.1)	0.297	0.279
(0,0)	0.039	0.053

Figure 6: Power Results Using Method of Separation

The results of this method show that it does a fairly good job at detecting mediation. These results are comparable to a single mediator if we just looked at one of the mediators Power as a reflection of the different coefficients. It appears that we have 100 percent power when the mediators have a bx coefficient ≥ 0.05 with n=200 and $\sigma = 1$. Another important aspect of this graph is that when there is no mediation present the power for both mediators is ≈ 0.05, which means we are getting plausible results.

Method=PCA Mediation (n=200, Simulations=1000, bm=c(1,1), σ=1)					
		Power	**Power**		
Bx	**#Reduced**	**PC.1 Reduced**	**#PCA**	**PC.1**	**PC.2**
(1,1)	968	1.00	32	1.00	0
(1,0.5)	430	1.00	570	1.00	0.0175
(1,0.2)	210	1.00	790	1.00	0.0519
(1,0.1)	166	1.00	834	1.0	0.0528
(0.5,0.1)	0	NA	1000	0	0.274
(0.5,0.5)	0	NA	1000	1.00	0.151
(0.2,0.2)	0	NA	1000	0.803	0.4
(0.1,0.1)	0	NA	1000	0.319	0.209
(0,0)	0	NA	1000	0.0420	0.0440

Figure 7: Power Results Using Method of Principal Component Analysis

The results of the PCA method are a little different. The way PCA works is that in order to reduce the dimensions the principal component is chosen so that the largest possible variance of the data can be displayed in the first principal component. Therefore, when two mediators that are fairly correlated end up using both principal components to meet the needed proportion of variance, most of the variance will be in the first PC, making rejection of the null for the second PC more difficult. If the mediators are correlated then using the 2'nd PC would not make much sense, especially considering there are only two mediators

here. As the strength of the mediation decreases the proportion of data sets that are reduced decreases as well. When they are reduced, however, the power is consistently 100 percent. When they are not, there is still obviously 100 percent power for the first PC but the second PC is generally close to 0. Similarly to the separate method, powers that are ≈ 0.05 are also found when there is no mediation present. Lower strength where bx< 0.05 tend to always need both principal components. Therefore, it would make sense that for these lower bx coefficients separation would be the best technique. This is why a combination function was created.

Method= Combined Method (n=200, Simulations=1000, bm=c(1,1), σ=1)								
	Power			**Power**			**Power**	
Bx	**#Reduced**	**PC.1.Reduced**	**#PCA**	**PC.1**	**PC.2**	**# Sep**	**M.1**	**M.2**
(1,1)	966	1	34	1	0	0	NA	NA
(1,0.5)	422	1	538	1	0.0167	40	1	1
(1,0.2)	78	1	107	1	0.0935	815	1	0.728
(1,0.1)	10	1	13	1	0.0769	977	1	0.266
(0.5,0.5)	4	1	493	1	0.0219	503	1	1
(0.2,0.2)	0	NA	19	1	0	981	0.794	0.778
(0.1,0.1)	0	NA	3	0.667	0	997	0.287	0.290
(0,0)	0	NA	2	0	0	998	0.0521	0.0401

Figure 8: Power Results Using Method of Combining Separation and PCA

The results in Figure 8 display for each combination of bx, how many data sets out of 1000 are reduced to one PC, how many need both PC's, and how many require separation of the mediators. We can see that when bx=c(1,1) we get almost exactly the same results as the PCA method. This makes sense, because if the mediators are highly correlated we are going to go into principal components instead of separation, and if the data very clearly have mediation present then the first PC will most likely meet the needed proportion of variance. This method seems to be a more realistic method that can be extended

to multiple mediators, because it will take out the mediators that are not correlated with the rest and can run principal component analysis on mediators that are correlated. As predicted from Figure 7 when the bx coefficient are < 0.05 the mediators are usually analyzed separately. Like before, when there is no mediation present, bx=(0,0), the powers are ≈ 0.05

Method=Combined Method(n=200, Simulations=100, bm=c(1,1,1), σ=1)					
		Power		Power	
Bx	#Reduced	PC.1	#Reduced with M.3 Sep	PC.1	M.3
(1,2,0.1)	66	1	934	1	0.210

Figure 9: Power Results Using Method of Combining Separation and PCA with Three Mediators Bx=(1,2,0.1)

Figure 9 gives us a better idea of when the combination function can be extremely useful. Instead of having three different p-values for each mediator the simulation resulted in two possibilities. The first result was that all three mediators were correlated enough to run PCA, and the first PC met the needed proportion of variance default of 0.7, which ended up with 66 cases where we would reject the null hypothesis that no mediation is present for all three mediators. The other result, which occurred 934 times, was have the first and second mediator go through PCA and analyzing the third mediator separately. This resulted in rejected the null hypothesis that there is no mediation present for the first and second mediators 100 percent of the time and rejected the null hypothesis that there is no mediation present for third mediator 21 percent of the time.

Figure 10 shows a more complicated result with three mediators. There is one strong mediator that comes from bx=1, and then a moderately strong mediator with bx=.5, and a weak mediator with bx=0.1. This produces a more complicated structure when simulated 1000 times. Out of the 1000 the mediators were reduced to two mediation analyses 456 times. Of those 456 times, 430 of them reduced the first two mediators to the first PC, and analyzed the third mediator separately. The other 26 times, PCA was run on all three mediators and then reduced to two mediators. Out of the other 544 times, 500 times times the data was reduced to PCA with the first two mediators and analyzed the third mediator separately. The three mediators were all analyzed separately 42 times out of the 544. Finally, there were 2 times that the

second and third mediator were used for PCA and the first mediator was analyzed separately.

Method=Combined Method (n=200, Simulations=100, bm=c(1,1,1), σ=1)																			
			Power				Power				Power				Power				
Bx	#Reduced	PC.1	M.3	#Reduced	PC.1	PC.2	#Reduced	M.1	M.2	M.3	#Reduced	PC.1	PC.2	M.1	#Reduced	PC.1	PC.2	M.3	
(1,0.5,.01)	430	1.0	0.28	26	1.00	0.044	42	1	1	0.26	2	1	0	1	500	1	0.012	0.29	

Figure 10: Power Results Using Method of Combining Separation and PCA with Three Mediators, Bx=(1,0.5,0.1)

4.2 Multiple Dependent Variables

To do this in R, simulating x and m remain the same except y is now being generated from a multivariate normal distribution to obtain y's that are based off of m and are correlated between each other. There also needs to be coefficients that are used to multiply by m to be used as the means for the multivariate y distribution. The mediate.sim function can create data set with multiple y's using the method="multivariate.y"

There are also multiple ways to run the mediation analysis using the correlated y's. One could use the first principle component, or any of the actual y's itself. The higher the proportion of the variance is in first principle component the better the prediction. The method used in the mediate.analyze function uses the first principal component prediction as the Y to use to test for mediation. Preliminary results comparing this method to just using one of the Y's from the multivariate normal distribution showed that this seemed to be promising when ρ is high.

5 Conclusion and Future Work

This paper ran power analysis to empirically show how well the Sobel Test does under certain conditions for mediation analysis with one mediator, one independent variable, and one dependent variable.The power analysis showed that changing bm by a certain amount had the same effect as changing bx by a certain amount, holding everything else constant. In other words, when looking at Figure 4 and Figure 5, both figures display the exact same trend. When n=200 and $\sigma = 1$, the bx needed to obtain near 100 percent power when bm ≈ 1 is around 0.3. When n=200 and $\sigma = 1$, the bm needed to obtain near 100 percent power when bx ≈ 1 is around 0.3 as well. These results were compared to bootstrapping and tested for robustness using Gamma error. After finding no significant differences, it seemed plausible to extend the idea of one mediator to multiple mediators using the Sobel Test.

After expanding into using multiple mediators, two initial ideas were explored. The first being analyzing each mediator separately. However, this created a multiple comparison issue that may be corrected through a Bonferroni correction, but with a loss of power. The second method was to run PCA on the

mediators to reduce the dimensions in which to test for mediation. However, having highly correlated mediators sometimes allow for a very low rejection percentage for higher principal components. Therefore, both of these methods were combined to create a new method for analyzing mediation. This method screens the mediators to take out and analyze separately any mediator that is not correlated with the rest, and this correlation level can be specified. The method then runs PCA and takes the number of PC predictions needed to obtain a minimum proportion of variance, and this can also be specified. Results from this analysis found that the combination method of analyzing some of the mediators separately and some of the more correlated mediators together using principal component analysis can achieve more successful results than the separate and PCA methods. Instead of performing a Bonferroni correction, the combined method reduces the number of mediators that are analyzed separately, thereby reducing the multiple comparisons problem. When comparing the combined method to the separate and PCA method it appears that it is more likely to choose a correct method given the effect sizes of the coefficients. If two mediators are very correlated and a third mediator doesn't appear to be strongly correlated then it makes sense to analyze the third mediator separately and run PCA on the two correlated mediators to determine whether or not mediation is present in these mediators. From Figure 9, one can see that this is what happens most of the time.

One issue with the simulations, however, is that looking at the results is both time consuming and not user friendly. All of the results were looked through on a case by case basis, and if further time was permitted this would be an issue to resolve. For each simulation it would be nice to create a function that could visualize these results in a comprehensible format. Possible methods of displaying these results in a graph might be worth trying to pursue. Of course, this issue does not apply to analysis of individual data sets. It only applies to someone trying to

The final section of the paper dealt with the idea of mediation with multiple dependent variables. However, the method presented in this paper and that is in the function simply runs PCA and uses the first PC as Y and runs mediation analysis as before. This method was compared to just using one of the Y's and they were found to have similar results. When trying to perform mediation analysis with multiple dependent variables, it may be beneficial to delve into other methods than these presented. One possibility

is to look into using MANOVA, Multivariate Analysis of Variance. If MANOVA was used to regress $Y \sim X$ and $Y \sim X + M$ then one could look at the overall p-values to determine if mediation is present using the Causal Steps method. If the overall p-value of X in the first equation is significant, and then in the second equation the coefficient has decreased or X the overall p-value is not significant then there is evidence of mediation. This method may cause not as great of a loss in the dependent variables as principal component analysis may cause. Future work might also want to look into cases where there are multiple dependent variables and multiple mediators, but this was beyond the scope of this project.

There could a great deal more analysis on mediation analysis with multiple mediators. One possibility is to determine if a Bonferroni correction could be used in the combined method in order to reduce the error associated with multiple comparisons. While, this problem may be reduced when running the combined method there are still multiple comparisons being made and this may be a possible solution. However, this would be a difficult task as the number of p-values produced are not predetermined before testing the data.

Future work on multiple mediation analysis may also be interested in mediation analysis with multiple mediators and additional independent variables. These additional variables would be in the equations used for mediation analysis, but would not detect from whether or not mediation is present. Arguments could be added to the function that allow for analyzed mediation with multiple mediators and more independent variables. Analysis would need to be done to see whether or not the independent variables had a significant effect on the mediation.

Appendix A

R Package: Multiple Mediation

After creating all of these function it made sense to put them together in a package for everyone to use. Thus, I created the package: MultipleMediaton. It consists of a function for creating simulated data to run mediation analysis, a function for finding the coefficients used for mediation analysis, a function that runs the mediation analysis, and finally a function that simulates and runs mediations analysis at the same time.

mediate.sim

The first function is called "Mediation.sim" The components of this function include: n, bx, bm, mean.x, sd.x, sd.m, sd.y, alpha, beta, method, and extras.The "n" is just to specify the number of data points that need to be simulated for each variable. "mean.x" and "sd.x" are the mean and standard deviation used to simulate the independent variable X from a normal distribution:

$$X \sim Normal(mean = mean.x, sd = sd.x)$$

If the method is "normal" then the function takes needs a bx and a bm to simulate M and Y:

$$M \sim Normal(mean = bx * x, sd = sd.m)$$

$$Y \sim Normal(mean = bm * m, sd = sd.y$$

If the method is "gamma" then the function takes an alpha and beta, in addition to bx and bm, to simulate M and Y using gamma error:

$$M \sim bx * x + Gamma(\alpha, \beta) - \alpha * \beta$$

$$Y \sim bm * m + Gamma(\alpha, \beta) - \alpha * \beta$$

If the method is "multivariate.m," takes in bx and bm that are vectors that must be of equal lengths. Then M's and Y are simulated as shown below:

$$M_i \sim Normal(mean = bx_i * X, sd = m.sd_i)$$

$$Y \sim Normal(mean = \sum(bm_i * M_i), sd = y.sd_i)$$

Finally, the last method is "multivariate.y," which is a little bit more complicated. Y now comes from a multivariate normal distribution based off of M, and the Y's are correlated between each other. There needs to be coefficients that are used to multiple by m to be used as the means for the multivariate y distribution, which can be specified by bm. The function will also need sds which are the standard deviations between the Y variables, as well as ρ and there can be one ρ for all of the Y's, or ρ can be different between different Y's. Then M's and Y are simulated as shown below (the Y is simulated with a rapper function):

$$M \sim Normal(mean = bx * x, sd = sd.m)$$

$$Y \sim MVN(mean = M(bm^T), \sigma = sds, \rho = \rho$$

mediate.analyze

The mediate.analyze function is what takes in a data set of X,M, and Y and gives back the mediated effect(s), SE(s) of the mediated effect(s), Z score(s) of mediated effect(s), and the p-value(s) of the mediated effect(s). The function consists of the arguments: data, method, which.y, need.prop.var,need.cor. The last three arguments are specific to certain methods.

The "normal" method uses a function "est.coef.y" to determine the a, b, $a * b$, se$_a b$, Z, and p-value. This method returns the singular p-value as its result. This method should be used when the data that was

simulated has 1 X, 1 M, and 1 Y.

The "multivariate.m.sep" method uses a function "est.coef.m.sep" to determine an a, b, $a * b$, se$_a b$, Z, and p-value for each mediator that is present in the data. Then the function will return the mediated effects, standard errors, Z's, and p-values for each mediator.

The "multivariate.m.pca" method uses a function "est.coef.pca" to determine how to reduce the number of dimensions in M using the needed proportion of variance for the principal component analysis. Then this function returns mediated effects, standard errors, Z's, and p-values that correspond to the principal component predictions used as a mediator to analyze.

The "multivariate.m" method combines both of the previous methods. It first looks at the correlations between the mediators, and if any of the mediators are uncorrelated with all the rest then that mediator will be analyzed separately. Then those that are correlated are then run through the PCA method and the number of dimensions to be reduced is found and we use those predictions in combination with the M's that are separated to determine an a, b, $a * b$, se$_a b$, Z, and p-value for all the m's that are separated and for the number principal component predictions needed.

sim.med.analyze

Finally, this function takes in all the arguments of mediate.sim as well as additional arguments used in mediate.analyze such as, method, need.prop.var, and need.cor. The function then simulates data based on the initial arguments and uses the latter arguments to run the mediate.analyze function to find the p-values.

Appendix B

R Code

```
my.rmvnorm=function(mu, sds, rho) {
k<-length(sds)
n=nrow(mu)
if (length(rho)==1) {
Cor=matrix(rho,k,k)
diag(Cor)=1
} else {
if (length(rho) != choose(k,2)) stop("wrong rho length")
Cor=diag(k)
Cor[lower.tri(Cor)]=rho
for (i in 1:(k-1)){
for (j in (i+1):k) {
Cor[i,j]=Cor[j,i]
}
}
}
Cov=diag(sds)%*%Cor%*%diag(sds)
wrapMVN= function(oneMu, Cov) {
mvrnorm(1,oneMu,Cov)
}
t(apply(mu,1,wrapMVN,Cov=Cov))
}

est.coef.y = function(data,which.y=0){
```

```
##browser()
n<-nrow(data)
##y.on.x<-lm(y~x,data=data)
y.on.x.m<-lm(y~x+m,data=data)
m.on.x<-lm(m~x,data=data)
##c.coef= coef(y.on.x)["x"]
##c.prime.coef=coef(y.on.x.m)["x"]
b.coef= coef(y.on.x.m)["m"]
a.coef = coef(m.on.x)["x"]
a.var<-coef(summary(m.on.x))["x","Std. Error"]^2
b.var<-coef(summary(y.on.x.m))["m","Std. Error"]^2
temp<-c(b.coef,a.coef,a.var,b.var)
names(temp)=c("b","a","a.var","b.var")
return(temp)
}

est.coef.pca.y = function(data,need.prop=0.7) {
cl <- match.call()
k<-ncol(data)-2
y.s<-data[,3:(k+2)]
ys.pca<-prcomp(y.s)
if((prop.var<-ys.pca$sdev[1]^2/sum(ys.pca$sdev^2)) < need.prop)
warning("Proportion of Variance is only ",prop.var)
y<-predict(ys.pca)[,1]
##y.on.x<-lm(y~x,data=data)
y.on.x.m<-lm(y~x+m,data=data)
m.on.x<-lm(m~x,data=data)
```

```
##c.coef= coef(y.on.x)["x"]
##c.prime.coef=coef(y.on.x.m)["x"]
b.coef= coef(y.on.x.m)["m"]
a.coef = coef(m.on.x)["x"]
a.var<-coef(summary(m.on.x))["x","Std. Error"]^2
b.var<-coef(summary(y.on.x.m))["m","Std. Error"]^2
names(a.coef)<-paste("a",1,sep="")
names(b.coef)<-paste("b",1,sep="")
names(a.var)<-paste("a.var",1,sep="")
names(b.var)<-paste("b.var",1,sep="")
rtn=list(call=cl,a.coef=a.coef,a.var=a.var,b.coef=b.coef,b.var=b.var,num.m=num
class(rtn)<-"med.model"
return(rtn)
}

est.coef.manova.y = function(data,which.y=1){
##browser()
k<-ncol(data)-2
y.s<-data[,3:(k+2)]
y<-y.s[,which.y]
##y.on.x<-lm(y~x,data=data)
y.on.x.m<-lm(y~x+m,data=data)
m.on.x<-lm(m~x,data=data)
##c.coef= coef(y.on.x)["x"]
##c.prime.coef=coef(y.on.x.m)["x"]
b.coef= coef(y.on.x.m)["m"]
a.coef = coef(m.on.x)["x"]
```

```
a.var<-coef(summary(m.on.x))["x","Std. Error"]^2
b.var<-coef(summary(y.on.x.m))["m","Std. Error"]^2
temp<-c(b.coef,a.coef,a.var,b.var)
names(temp)=c("b","a","a.var","b.var")
return(temp)
}
est.coef.sep = function(data,which.y=0,verbose=FALSE){
##browser()
cl <- match.call()
n<-nrow(data)
num.m<-ncol(data)-2
##y.on.x<-lm(y~x,data=data)
form<-paste("y~x+",paste('m', 1:num.m,sep=".",collapse="+"))
y.on.x.m<-lm(as.formula(form),data=data)
if(verbose) print(summary(y.on.x.m))
a.coef=a.var=rep(NA,num.m)
for (i in 1:num.m) {
form<-paste(paste('m', i,sep="."),"~x")
temp<-lm(as.formula(form),data=data)
a.coef[i]<-coef(temp)["x"]
a.var[i]<-coef(summary(temp))["x","Std. Error"]^2
if(verbose) print(summary(temp))
}
##c.coef= coef(y.on.x)["x"]
##c.prime.coef=coef(y.on.x.m)["x"]
b.coef= coef(y.on.x.m)[paste('m', 1:num.m,sep=".")]
b.var<-coef(summary(y.on.x.m))[paste('m', 1:num.m,sep="."),"Std. Error"]^2
```

```
names(a.coef)<-paste("a",1:num.m,sep="")
names(b.coef)<-paste("b",1:num.m,sep="")
names(a.var)<-paste("a.var",1:num.m,sep="")
names(b.var)<-paste("b.var",1:num.m,sep="")
rtn=list(call=cl,a.coef=a.coef,a.var=a.var,b.coef=b.coef,b.var=b.var,num.m=num
class(rtn)<-"med.model"
return(rtn)
}

est.coef.pca<-function(data,need.prop.var=0.7,need.corr=0.2,verbose=FALSE) {
cl <- match.call()
k<-ncol(data)
x<-data[,1]
y<-data[,k]
m.s<-data[,2:(k-1)]
num.m<-ncol(m.s)
ms.pca<-prcomp(m.s)
prop.var<-cumsum(ms.pca$sdev^2)/sum(ms.pca$sdev^2)
if(prop.var[1] > need.prop.var) {
pc.a<-predict(ms.pca)[,1]
##y.on.x<-lm(y~x,data=data)
y.on.x.m<-lm(y~x+pc.a,data=data)
m.on.x<-lm(pc.a~x,data=data)
##c.coef= coef(y.on.x)["x"]
##c.prime.coef=coef(y.on.x.m)["x"]
b.coef= coef(y.on.x.m)["pc.a"]
a.coef = coef(m.on.x)["x"]
```

```
a.var<-coef(summary(m.on.x))["x","Std. Error"]^2
b.var<-coef(summary(y.on.x.m))["pc.a","Std. Error"]^2
names(a.coef)<-paste("a",1,sep="")
names(b.coef)<-paste("b",1,sep="")
names(a.var)<-paste("a.var",1,sep="")
names(b.var)<-paste("b.var",1,sep="")
rtn=list(call=cl,a.coef=a.coef,a.var=a.var,b.coef=b.coef,b.var=b.var,num.m=1)
class(rtn)<-"med.model"
return(rtn)
} else  {

pc.needed<-1+length(which(prop.var<=need.prop.var))
m.s<-m.s[,1:pc.needed]
for (i in 1:pc.needed) {
data[,i+1]<-predict(ms.pca)[,i]
}
form<-paste("y~x+",paste('m', 1:pc.needed,sep=".",collapse="+"))
y.on.x.m<-lm(as.formula(form),data=data)
if(verbose) print(summary(y.on.x.m))
a.coef=a.var=rep(NA,num.m)
for (i in 1:pc.needed) {
form<-paste(paste('m', i,sep="."),"~x")
temp<-lm(as.formula(form),data=data)
a.coef[i]<-coef(temp)["x"]
a.var[i]<-coef(summary(temp))["x","Std. Error"]^2
if(verbose) print(summary(temp))
}
```

```
##c.coef= coef(y.on.x)["x"]
##c.prime.coef=coef(y.on.x.m)["x"]
b.coef= coef(y.on.x.m)[paste('m', 1:pc.needed,sep=".")]
b.var<-coef(summary(y.on.x.m))[paste('m', 1:pc.needed,sep="."),"Std. Error"]^2
names(a.coef)<-paste("a",1:pc.needed,sep="")
names(b.coef)<-paste("b",1:pc.needed,sep="")
names(a.var)<-paste("a.var",1:pc.needed,sep="")
names(b.var)<-paste("b.var",1:pc.needed,sep="")
rtn=list(call=cl,a.coef=a.coef,a.var=a.var,b.coef=b.coef,b.var=b.var,num.m=pc.
class(rtn)<-"med.model"
return(rtn)
}
}



est.coef.m.det = function(data,need.prop.var=0.7,need.cor=0.2,verbose=FALSE,..
cl <- match.call()
k<-ncol(data)
x<-data[,1]
y<-data[,k]
m.s<-data[,2:(k-1),drop=FALSE]
num.m<-ncol(m.s)
##if one of the m's is not correlated with the rest then that m should be anal
##separately

cors.less<-cor(m.s)< need.cor
```

```
tf.ms<-apply(cors.less,1,sum)==num.m-1
if(any(tf.ms)==TRUE) {
ms.sep<-m.s[,tf.ms,drop=FALSE]
}  else {
ms.sep<-NULL
}
if(any(tf.ms==FALSE)) {
ms.new<-m.s[,tf.ms==FALSE,drop=FALSE]
colnames(ms.new)<-gsub("m","pc.m",colnames(ms.new),fixed=TRUE)
new.pca<-prcomp(ms.new)
prop.var<-cumsum(new.pca$sdev^2)/sum(new.pca$sdev^2)
if(prop.var[1] > need.prop.var) {
ms.red<-predict(new.pca)[,1,drop=FALSE]
colnames(ms.red)<-paste("pc.a")
}   else {
pc.needed<-1+length(which(prop.var<=need.prop.var))
ms.red<-ms.new[,1:pc.needed]
for (i in 1:pc.needed) {
ms.red[,i]<-predict(new.pca)[,i]
}
colnames(ms.red)<-paste("pc.",letters[1:pc.needed],sep="")
}
}  else {
ms.red<-NULL
}
if( is.null(ms.sep)) {
all.ms<-ms.red
```

```
} else if(is.null(ms.red)){
all.ms<-ms.sep
} else {
all.ms<-cbind(ms.red,ms.sep)
}
all.ms<-as.data.frame(all.ms)
m.col<-ncol(all.ms)
all.ms<-cbind(all.ms,x,y)

form<-paste("y~x+",paste(colnames(all.ms)[1:m.col],collapse="+"))
y.on.x.m<-lm(as.formula(form),data=all.ms)
if(verbose) print(summary(y.on.x.m))
a.coef=a.var=rep(NA,m.col)
b.coef=b.var=rep(NA,m.col)
for (i in 1:m.col) {
form<-paste(colnames(all.ms)[i],"~x")
temp<-lm(as.formula(form),data=all.ms)
a.coef[i]<-coef(temp)["x"]
a.var[i]<-coef(summary(temp))["x","Std. Error"]^2
b.coef[i]= coef(y.on.x.m)[paste(colnames(all.ms)[i])]
b.var[i]<-coef(summary(y.on.x.m))[paste(colnames(all.ms)[i]),"Std. Error"]^2
if(verbose) print(summary(temp))
}
##c.coef= coef(y.on.x)["x"]
##c.prime.coef=coef(y.on.x.m)["x"]
temp<-c(b.coef,a.coef,b.var,a.var)
```

```
n.sep<-ifelse(is.null(ms.sep),0,ncol(ms.sep))
n.red<-ifelse(is.null(ms.red),0,ncol(ms.red))

names(a.coef)<-paste("a",1:m.col,sep="")
names(b.coef)<-paste("b",1:m.col,sep="")
names(a.var)<-paste("a.var",1:m.col,sep="")
names(b.var)<-paste("b.var",1:m.col,sep="")
rtn=list(call=cl,a.coef=a.coef,a.var=a.var,b.coef=b.coef,b.var=b.var,num.m=num
class(rtn)<-"med.model"
return(rtn)
 }



mediate.sim<-function(n,bx,bm,mean.x=0,sd.x=1,sd.m=1,sd.y=1,alpha,beta,method=

method=match.arg(method)
if(method=="normal") {
x<-rnorm(n,mean.x,sd.x)
m<-rnorm(n,mean=bx*x,sd=sd.m)
y<-rnorm(n,mean=bm*m,sd=sd.y)
sim.data<-cbind(x,m,y)
return(data.frame(sim.data))
} else if(method=="gamma") {
x<-rnorm(n,mean.x,sd.x)
m<-bx*x + rgamma(n,alpha,1/beta) - alpha*beta
y<-bm*m + rgamma(n,alpha,1/beta) - alpha*beta
```

```
sim.data<-cbind(x,m,y)
return(data.frame(sim.data))
}else if(method=="multivariate.m") {
x<-rnorm(n,mean.x,sd.x)
num.m<-length(bx)
if(length(bx) != length(bm)) {
stop("Length of bx must equal length of bm")
}
m<-matrix(rnorm(n*num.m,mean=rep(bx,each=n)*x,sd=sd.m),
nrow=n,ncol=num.m)
colnames(m) <- paste('m', 1:num.m,sep=".")
diag.mat<-diag(bm)
mat.for.y<-m%*%diag.mat
mean.y<-apply(mat.for.y,1,sum)
y<-rnorm(n,mean=mean.y,sd=sd.y)
sim.data<-cbind(x,m,y)
return(data.frame(sim.data))
}else if(method=="multivariate.y") {
k<-length(bm)
if(length(extra$sds)==1) extra$sds=rep(extra$sds,k)
if (length(bm) != length(extra$sds)) stop("bm doesn't match sds")
b0=extra$b[1]
bx=extra$b[2]
x<-rnorm(n,mean.x,sd.x)
m<-rnorm(n,mean=b0+bx*x,sd.m)
mu.s<-m%*%t(bm)
y.s<-my.rmvnorm(mu=mu.s,sds=extra$sds,rho=extra$rho)
```

```
colnames(y.s)=paste("y",1:k,sep="")
sim.data<-cbind(x,m,y.s)
return(data.frame(sim.data))
}
}
try<-mediate.sim(n=100,bm=c(1,2),extra=list(b=c(0,1),sds=c(.4,.7),rho=.5),meth
##s3 classes



mediate.analyze<-function(data,verbose=FALSE,which.y=0,method=c("normal","mult
cl <- match.call()
if(method=="normal") {
coef<-est.coef.y(data,which.y)
##c<-coef["c"]
##c.prime<-coef["emirp.c"]
b<-coef["b"]
a<-coef["a"]
a.var<-coef["a.var"]
b.var<-coef["b.var"]
prod.me<-a*b
##diff.me<-c- c.prime
var.diff<-a.var*b^2 + b.var*a^2
se.me<-var.diff^.5
z<-prod.me/se.me
p.zvalue<-2*pnorm(-abs(z))
if(verbose) return(list(prod.me=prod.me,se.me=se.me,  z=z,p.zvalue=p.zvalue))
```

```
return(as.numeric(p.zvalue))
} else if (method=="multivariate.m") {
coef<-est.coef.m.det(data,need.prop.var=0.7,need.cor=0.2)
num.med<-length(coef$which.m.s)
##c<-coef["c"]
##c.prime<-coef["emirp.c"]
a<-coef$a.coef
b<-coef$b.coef
a.var<-coef$a.var
b.var<-coef$b.var
prod.me<-a*b
names(prod.me)<-paste("ab",1:num.med,sep="")
##diff.me<-c- c.prime
var.diff<-a.var*b^2 + b.var*a^2
se.me<-var.diff^.5
names(se.me)<-paste("se.ab",1:num.med,sep="")
z<-prod.me/se.me
names(z)<-paste("z",1:num.med,sep="")
p.zvalue<-2*pnorm(-abs(z))
p.ms<-c(as.numeric(p.zvalue))
names(p.ms)<-paste("pval",coef$which.m.s,sep=".")
rtn=list(call=cl,prod.me=prod.me,se.me=se.me,z=z,p.ms=p.ms)
class(rtn)<-"med.result"
return(rtn)
} else if (method=="multivariate.m.sep") {
coef<-est.coef.sep(data,verbose=FALSE)
num.med<-length(coef$a.coef)
```

```
##c<-coef["c"]
##c.prime<-coef["emirp.c"]
a<-coef$a.coef
b<-coef$b.coef
a.var<-coef$a.var
b.var<-coef$b.var
prod.me<-a*b
names(prod.me)<-paste("ab",1:num.med,sep="")
##diff.me<-c- c.prime
var.diff<-a.var*b^2 + b.var*a^2
se.me<-var.diff^.5
names(se.me)<-paste("se.ab",1:num.med,sep="")
z<-prod.me/se.me
names(z)<-paste("z",1:num.med,sep="")
p.zvalue<-2*pnorm(-abs(z))
p.ms<-c(as.numeric(p.zvalue))
names(p.ms)<-paste("pval",letters[1:num.med],sep=".")
rtn=list(call=cl,prod.me=prod.me,se.me=se.me,z=z,p.ms=p.ms)
class(rtn)<-"med.result"
return(rtn)
} else if(method=="multivariate.m.pca") {
coef<-est.coef.pca(data)
num.med<-length(coef$a.coef)
##c<-coef["c"]
##c.prime<-coef["emirp.c"]
a<-coef$a.coef
b<-coef$b.coef
```

```
a.var<-coef$a.var
b.var<-coef$b.var
prod.me<-a*b
names(prod.me)<-paste("ab",1:num.med,sep="")
##diff.me<-c- c.prime
var.diff<-a.var*b^2 + b.var*a^2
se.me<-var.diff^.5
names(se.me)<-paste("se.ab",1:num.med,sep="")
z<-prod.me/se.me
names(z)<-paste("z",1:num.med,sep="")
p.zvalue<-2*pnorm(-abs(z))
p.ms<-c(as.numeric(p.zvalue))
names(p.ms)<-paste("pval",1:num.med,sep=".")
rtn=list(call=cl,prod.me=prod.me,se.me=se.me,z=z,p.ms=p.ms)
class(rtn)<-"med.result"
return(rtn)
} else if(method=="multivariate.y")

  coef<-est.coef.pca.y(data)
  a<-coef$a.coef
  b<-coef$b.coef
a.var<-coef$a.var
b.var<-coef$b.var
prod.me<-a*b
names(prod.me)<-paste("ab1")
##diff.me<-c- c.prime
var.diff<-a.var*b^2 + b.var*a^2
```

```
se.me<-var.diff^.5
names(se.me)<-paste("se.ab1")
z<-prod.me/se.me
names(z)<-paste("z1")
p.ms<-2*pnorm(-abs(z))
names(p.ms)<-paste("p1")
rtn=list(call=cl,prod.me=prod.me,se.me=se.me,z=z,p.ms=p.ms)
class(rtn)<-"med.result.manova"
return(rtn)
}


sim.med.analyze<-function(n,bx,bm,mean.x=0,sd.x=1,sd.m=1,sd.y=1,alpha,beta,met
data<-
mediate.sim(n,bx,bm,mean.x,sd.x,sd.m,sd.y,alpha,beta,method=method.sim)
temp<-
mediate.analyze(data,method=method.analyze,need.prop.var,need.cor)
}
Picture 107.pngPicture 106.png
summary.med.result.manova<-function (object, correlation = FALSE, symbolic.cor
    ...)
{
    class(object) <- "summary.med.result.manova"
    object
}
```

```
print.med.result.manova<-function (x, digits = max(3, getOption("digits") - 3)
{
 cat("\nCall:\n", paste(deparse(x$call), sep = "\n", collapse = "\n"),
        "\n\n", sep = "")

        cat("Mediated Effect Coefficient: \n")
        print.default(format(x$prod.me, digits = digits), print.gap = 2,
            quote = FALSE)
        cat("\n")
        cat("Mediated Effect Standard Error:\n")
        print.default(format(x$se.me, digits = digits), print.gap = 2,
            quote = FALSE)
        cat("\n")
        cat("Z Score of Mediated Effect:\n")
        print.default(format(x$z, digits = digits), print.gap = 2,
            quote = FALSE)
        cat("\n")
        cat("P-value of Mediated Effect:\n")
        print.default(format(x$p.ms, digits = digits), print.gap = 2,
            quote = FALSE)
        cat("\n")

    invisible(x)

 }
```

```
summary.med.result<-function (object, correlation = FALSE, symbolic.cor = FALS
    ...)
{
    class(object) <- "summary.med.result"
    object
}


print.med.result<-function (x, digits = max(3, getOption("digits") - 3), ...)
{
 cat("\nCall:\n", paste(deparse(x$call), sep = "\n", collapse = "\n"),
        "\n\n", sep = "")

        cat("Mediated Effect Coefficients: \n")
        print.default(format(x$prod.me, digits = digits), print.gap = 2,
            quote = FALSE)
        cat("\n")
        cat("Mediated Effect Standard Errors:\n")
        print.default(format(x$se.me, digits = digits), print.gap = 2,
            quote = FALSE)
        cat("\n")
        cat("Z Scores of Mediated Effects:\n")
        print.default(format(x$z, digits = digits), print.gap = 2,
            quote = FALSE)
        cat("\n")
        cat("P-values of Mediated Effects:\n")
```

```
        print.default(format(x$p.ms, digits = digits), print.gap = 2,
            quote = FALSE)
        cat("\n")

    invisible(x)

 }


print.med.model<-function (x, digits = max(3, getOption("digits") - 3), ...)
{
    cat("\nCall:\n", paste(deparse(x$call), sep = "\n", collapse = "\n"),
        "\n\n", sep = "")

        cat("Coefficients on X:\n")
        print.default(format(x$a.coef, digits = digits), print.gap = 2,
            quote = FALSE)
        cat("\n")
        cat("Coefficients on M:\n")
        print.default(format(x$b.coef, digits = digits), print.gap = 2,
            quote = FALSE)
        cat("\n")
        cat("Coefficients on X Variance:\n")
        print.default(format(x$a.var, digits = digits), print.gap = 2,
            quote = FALSE)
        cat("\n")
        cat("Coefficients on M Variance:\n")
```

```
        print.default(format(x$b.var, digits = digits), print.gap = 2,
            quote = FALSE)
        cat("\n")

    invisible(x)
}


summary.med.model<-function (object, correlation = FALSE, symbolic.cor = FALSE
    ...)
{
    class(object) <- "summary.med.model"
    object
}

print.med.model.summary<-function (x, digits = max(3, getOption("digits") - 3)
{
    cat("\nCall:\n", paste(deparse(x$call), sep = "\n", collapse = "\n"),
        "\n\n", sep = "")

        cat("Coefficients on X:\n")
        print.default(format(x$a.coef, digits = digits), print.gap = 2,
            quote = FALSE)
        cat("\n")
        cat("Coefficients on M:\n")
        print.default(format(x$b.coef, digits = digits), print.gap = 2,
            quote = FALSE)
```

```
        cat("\n")
        cat("Coefficients on X Variance:\n")
        print.default(format(x$a.var, digits = digits), print.gap = 2,
            quote = FALSE)
        cat("\n")
        cat("Coefficients on M Variance:\n")
        print.default(format(x$b.var, digits = digits), print.gap = 2,
            quote = FALSE)
        cat("\n")

    invisible(x)
}
```

References

Abdi, Herv, and Lynne J. Williams. ”Principal Component Analysis.” Wiley Interdisciplinary Reviews: Computational Statistics 2.4 (2010): 433-59. Print.

Carey, Gregory. ”Multivariate Analysis of Variance (MANOVA): I. Theory.” Colorado.edu. University of Colorado Boulder, 1998. Web. 04 Nov. 2011.

Haase, Richard F., and Michael V. Ellis. ”Multivariate Analysis of Variance.” Journal of Counseling Psychology 34.4 (1987): 404-13. Print.

Hayes, Andrew. ”Beyond Baron and Kenny: Statistical Mediation Analysis in the New Millennium.” Communication Monographs 76.4 (2009): 408-20. Print.

Imai, Kosuke, Luke Keele, and Dustin Tingley. ”A General Approach to Causal Mediation Analysis.” Psychological Methods (2010). Print.

Imai, Kosuke, Luke Keeler, Dustin Tingley, and Tepperi Yamamoto. ”Causal Mediation Analysis Using R.” Cran.r-project.org. 18 Sept. 2011. Web. 26 Jan. 2012.

Judd, Charles M., David A. Kenny, and Gary H. McClelland. ”Estimating and Testing Mediation and Moderation in Within-subject Designs.” Psychological Methods 6.2 (2001): 115-34. Print.

Kenny, David A. ”Mediation.” Davidakenny.net. 3 Apr. 2012. Web. 6 Apr. 2012.

MacKinnon, David P., Amanda J. Fairchild, and Matthew S. Fritz. ”Mediation Analysis.” Annual Review of Psychology 58.1 (2007): 593-614. Print.