

11-1-2016

Monte Carlo Simulations For Structural Equation Modelling (Revolution R)


Sarah A. Rose

Wayne State University, rosey13579@gmail.com

Barry Markman

Wayne State University, barry.markman@wayne.edu

Follow this and additional works at: <http://digitalcommons.wayne.edu/jmasm>

 Part of the [Applied Statistics Commons](#), [Social and Behavioral Sciences Commons](#), and the [Statistical Theory Commons](#)

Recommended Citation

Rose, Sarah A. and Markman, Barry (2016) "Monte Carlo Simulations For Structural Equation Modelling (Revolution R)," *Journal of Modern Applied Statistical Methods*: Vol. 15: Iss. 2, Article 43.

DOI: [10.22237/jmasm/1478004060](https://doi.org/10.22237/jmasm/1478004060)

Available at: <http://digitalcommons.wayne.edu/jmasm/vol15/iss2/43>

This Algorithms and Code is brought to you for free and open access by the Open Access Journals at DigitalCommons@WayneState. It has been accepted for inclusion in Journal of Modern Applied Statistical Methods by an authorized administrator of DigitalCommons@WayneState.

Monte Carlo Simulations for Structural Equation Modelling (Revolution R)

Sarah A. Rose
Wayne State University
Detroit, MI

Barry Markman
Wayne State University
Detroit, MI

Revolution R code is presented to setup Structural Equation Model (SEM) for a Monte Carlo study. The example is a comparison of different fit indices.

Keywords: Revolution R, structural equation model, SEM, fit indices, RMSEA, SRMR, CFI, chi-squared

Revolution R Lavaan package contains algorithms for performing SEM analytics. It has the ability to perform Monte Carlo simulations. The example considered here is for comparing model fit indices. This algorithm was verified using two well-known SEM computer software programs by extracting the last repetition of the Monte Carlo simulation in Revolution R and comparing the output results with IBM SPSS Amos Graphics and with Mplus Version 5.1.

The algorithm was developed for a 4×4 correlation matrix of random values constrained within a specific range. Lines 4 and 5 of the code indicate the minimum and maximum values of the correlation value range, respectively. These values are defined as variable “b1” for the lower limit and “c1” for the upper limit. Line 6, variable “var”, specifies the variant range that increases in magnitude based on the location of the correlation value within the correlation matrix. The algorithm defaults the variant range to zero, so all random correlation values for the matrix are within the same range.

The SEM model was designed for four variables: X_1 , X_2 , X_3 , and X_4 . The first three are exogenous variables, and X_4 is the only endogenous variable. The 4×4 correlation matrix is specified on line 75. To modify the correlation matrix to

Dr. Rose is an Adjunct Instructor of Education Evaluation and Research. Email her at: ak1734@wayne.edu. Dr. Markman is a Professor of Educational Psychology and Education Evaluation and Research. Email him at: barry.markman@wayne.edu.

another matrix size (i.e. 3×3 matrix, or 5×5 matrix) the variables on lines 7 through 16 would need to be redefined. Furthermore, the SEM model specification within the Monte Carlo command loop would require appropriate modifications on lines 69 through 80.

The model is specified with no correlation relationships, and no direct paths between variables X_1 and X_2 , X_1 and X_3 , and X_2 and X_3 . Relationships can be added or modified by deleting or modifying lines 78 through 80, which are currently designed to force relationship values to zero.

The algorithm was designed to provide six output files for six sample sizes (50, 100, 150, 200, 300, and 500). To change the sample sizes, modify the variable names and values on lines 18 through 23. The new variable names will have to be modified accordingly on lines 82 and 140.

The code provided in this article only contains the Monte Carlo simulation for one sample size of 50. Repetition of the algorithm logic for the other five sample sizes has been deleted to more concisely display the algorithm in this article. The repeated code should be copied and pasted, as appropriate, for compilation of all six output files. To do this, copy lines 68 through 156 for each of the various sample sizes required. The variable “ss50” should be replaced with the appropriate variable names that were specified in lines 19 through 23 (e.g. “ss100”, “ss150”, “ss200”, “ss300”, and “ss500”). These variable names should replace “ss50” in the copied version of lines 82, 140, and 156.

To modify the number of repetitions for the Monte Carlo simulation loop, modify the variable value of “rep” on line 25.

The model fit indices calculated and provided in the output files are the Chi-Squared (“baseline.pvalue” as specified by Revolution R), Root Mean Square Error Approximation (RMSEA), Standardized Root Mean Square Residual (SRMR), and Comparative Fit Index (CFI). The output files indicate the sample size, number of repetitions, mean degrees of freedom for all simulations, specified correlation range, and the percentage of times the SEM resulted in model fit index values above or below a critical value.

The Lavaan Package contains additional model fit indices that can be added to the output file. These include Tucker-Lewis Index (TLI), Goodness of Fit Index (GFI), Adjusted Goodness of Fit Index (AGFI), Normalized Fit Index (NFI), and Nonnormalized Fit Index (NNFI). To add these fit indices, copy lines 98 through 102 and paste them at the end of the repetition loop. Replace references to “srmr” with “tli”, “gfi”, “agfi”, “nfi”, or “nnfi”, as appropriate. Prepare the output by copying and modifying lines 118 through 120, lines 133 and 134, and lines 148 and 149 appropriately. The new variables created for these model fit indices would

MONTE CARLO SIMULATIONS FOR SEM (REVOLUTION R)

require specification of variable length. Insert the appropriate length and zero constant values, after lines 39 and 66, by copying the previous lines and modifying the variable names as appropriate.

Figure 1. R code and Lavaan package for conducting SEM fit indices

```
#Load SEM Lavaan Package
library(lavaan)
#Specify the Correlation Matrix Value Range
b1=0.06
c1=0.07
var=0
b2=b1+var
c2=c1+var
b3=b2+var
c3=c2+var
b4=b3+var
c4=c3+var
b5=b4+var
c5=c4+var
b6=b5+var
c6=c5+var
#set sampe size
ss50=50
ss100=100
ss150=150
ss200=200
ss300=300
ss500=500
#set number of repetition
rep=1000
#create arrays of specified length
reja5=numeric(length=rep)
reja1=numeric(length=rep)
reja01=numeric(length=rep)
acceptb=numeric(length=rep)
rejectb=numeric(length=rep)
acceptc=numeric(length=rep)
rejectc=numeric(length=rep)
acceptd=numeric(length=rep)
rejectd=numeric(length=rep)
close9=numeric(length=rep)
close75=numeric(length=rep)
close5=numeric(length=rep)
dof=numeric(length=rep)
#initialize constants to zero
totreja5=NULL
```

ROSE & MARKMAN

```

totreja1=NULL
totreja01=NULL
totacceptb=NULL
totrejectb=NULL
totacceptc=NULL
totrejectc=NULL
totacceptd=NULL
totrejectd=NULL
totlessthane9=NULL
ttotlessthane75=NULL
totlessthane5=NULL
totdof=NULL
tle5a=NULL
tle1a=NULL
tle01a=NULL
tleaccb=NULL
tlerejb=NULL
tleacc= NULL
tlerejc=NULL
tleaccd=NULL
tlerejd=NULL
tlecl9=NULL
tlecl75=NULL
tlecl5=NULL
meandof=NULL
#start loop
for (i in 1:rep) {
  d1=runif(1,b1,c1)
  d2=runif(1,b2,c2)
  d3=runif(1,b3,c3)
  d4=runif(1,b4,c4)
  d5=runif(1,b5,c5)
  d6=runif(1,b6,c6)
  data.cor=lav_matrix_lower2full(c(1,d1,1,d2,d3,1,d4,d5,d6,1))
  rownames(data.cor)=colnames(data.cor)=c("z","x1","x2","x3")
  model='z~x1+x2+x3
  x1~~0*x2
  x1~~0*x3
  x2~~0*x3
  '
  fit=sem(model,sample.cov=data.cor,sample.nobs=ss50,fixed.x=FALSE)
  doff=fitMeasures(fit,"df")
  dof[i]<-doff
  chisqpvalue=fitMeasures(fit,"baseline.pvalue")
# If chisqpvalue is significant at various levels, increment counter
  if (chisqpvalue <= .05) reja5[i] <- 1
  if (chisqpvalue > .05) reja1[i] <- 1
  if (chisqpvalue < .001) reja01[i] <- 1
  rmsealower=fitMeasures(fit,"rmsea.ci.lower")
}

```

MONTE CARLO SIMULATIONS FOR SEM (REVOLUTION R)

```
# If RMSEA value is less than 0.05
  if (rmsealower <= .05) acceptb[i] <- 1
  if (rmsealower > .05) rejectb[i] <- 1
  rmsealower=fitMeasures(fit,"rmsea.ci.upper")
# If rmsealower value is less than 0.1
  if (rmsealower <= .1) acceptc[i] <- 1
  if (rmsealower > .1) rejectc[i] <- 1
  srmr=fitMeasures(fit,"srmr")
# If srmr value is less than 0.09
  if (srmr <= .09) acceptd[i] <- 1
  if (srmr > .09) rejectd[i] <- 1
  cfi=fitMeasures(fit,"cfi")
# If cfi value is close to 1.0
  if (cfi < .9) close9[i] <- 1
  if (cfi < .75) close75[i] <- 1
  if (cfi >= .9) close5[i] <- 1
}
#A sum the number of rejections
totreja5=sum(reja5)
totreja1=sum(reja1)
totreja01=sum(reja01)
#B sum the number of rejections
totacceptb=sum(acceptb)
totrejectb=sum(rejectb)
#C sum the number of rejections
totacceptc=sum(acceptc)
totrejectc=sum(rejectc)
#D sum the number of rejections
totacceptd=sum(acceptd)
totrejectd=sum(rejectd)
#E sum the number of rejections
totlessthane9=sum(close9)
totlessthane75=sum(close75)
totlessthane5=sum(close5)
totdof=sum(dof)
#divide the sum by number of repetitions
tle5a=totreja5/rep
tle1a=totreja1/rep
tleaccb=totacceptb/rep
tlerejb=totrejectb/rep
tleacc= totacceptc/rep
tlerejc=totrejectc/rep
tleaccd=totacceptd/rep
tlerejd=totrejectd/rep
tlecl9=totlessthane9/rep
tlecl75=totlessthane75/rep
tlecl5=totlessthane5/rep
meandof=totdof/rep
# Summarize results in output file
```

ROSE & MARKMAN

```
results <- c("sample size=",ss50,"repetitions=",rep,
  "mean DOF=",meandof, "correlation=",b1,"-",c1,
  "chisq a<=0.05=",tle5a,"-> This means that p-chi-squared was less
  than 0.05 % of the time",
  "chisq a>0.05=",tle1a,"-> This means that p-chi-squared was
  greater than 0.05 % of the time",
  "RMSEA (lower)<=0.05=",tleaccb,"-> This means that RMSEA lower is
  less than 0.05 % of the time",
  "RMSEA (lower)>0.05=",tlerejb,"-> This means that RMSEA lower is
  greater than 0.05 % of the time",
  "RMSEA (upper)<=0.1=",tleaccc,"-> This means that RMSEA upper is
  less than 0.1 % of the time",
  "RMSEA (upper)>0.1=",tlerejc,"-> This means that RMSEA upper is
  greater than 0.1 % of the time",
  "SRMR<=0.09=",tleaccd,"-> This means that SRMR is less than
  0.09 % of the time",
  "SRMR>0.09=",tlerejd,"-> This means that SRMR is greater than
  0.09 % of the time",
  "CFI<0.9=",tlecl9,"-> This means that CFI is less than 0.9 % of
  the time",
  "CFI<0.75=",tlecl75,"-> This means that CFI is less than 0.75 %
  of the time",
  "CFI>=0.9=",tlecl5,"-> This means that CFI is greater than 0.9 %
  of the time")
# Write results to the hard disk.
# To write the results to your computer
# change file path to the folder you created on the C drive.
cat(results,sep="\n",file="c:/Users/Sarah/2cor_results-4variables-
50.txt",append=TRUE)#
#
#
#Repeat Lines 68 through 156 with appropriate modifications of variable
#“ss50” on lines 82 & 140 and output file name on line 156.
```