Education in Medicine Journal

EDUCATIONAL RESOURCE

Title: A Web-Based Sample Size Calculator for Structural Equation Modeling

Authors: Wan Nor Arifin

Submitted Date: 07-10-2024

Accepted Date: 01-11-2024

Please cite this article as: Arifin WA. A web-based sample size calculator for structural equation modeling. Education in Medicine Journal. (early view).

This is a provisional PDF file of an article that has undergone enhancements after acceptance, such as the addition of a cover page and metadata, and formatting for readability, but it is not yet the definitive version of record. This version will undergo additional copyediting, typesetting and review before it is published in its final form, but we are providing this version to give early visibility of the article.

EDUCATIONAL RESOURCE

ARTICLE INFO

Submitted: 07-10-2024
Accepted: 01-11-2024

A Web-Based Sample Size Calculator for Structural Equation Modeling

Wan Nor Arifin

Biostatistics and Research Methodology Unit, School of Medical Sciences, Universiti Sains Malaysia, Kelantan, Malaysia

ABSTRACT

Planning studies involving confirmatory factor analysis (CFA) and structural equation modeling (SEM) requires determining adequate sample sizes. Available methods for this include rule-of-thumb, Monte Carlo simulation, and sample size formulas. Manual calculations using sample size formulas are tedious and prone to errors, making software-based solutions preferable. This article introduces a user-friendly, web-based calculator for sample size determination in CFA and SEM studies. The calculator utilizes established formulas based on the root mean squared error of approximation and comparative fit index. The development process and core functionalities are discussed, along with demonstrations using common CFA and SEM examples. Additionally, I compare this calculator with other available web-based sample size calculators for SEM.

Keywords: *confirmatory factor analysis, sample size calculator, structural equation modeling, web-based software*

CORRESPONDING

Wan Nor Arifin, Biostatistics and Research Methodology Unit, School of Medical Sciences, Universiti Sains Malaysia, 16150 Kubang Kerian, Kelantan, Malaysia

Email: wnarifin@usm.my

INTRODUCTION

Planning a study involving structural equation modeling (SEM) and its measurement model via confirmatory factor analysis (CFA) requires the determination of an adequate number of respondents to ensure an acceptable level of precision and statistical power of parameter estimates, and reliable model fit indices (1). Available methods are rules of thumb (2), Monte Carlo simulation (3) and sample size formula (4). Hand calculation using sample size formulas is tedious and error-prone, thus, software-based sample size calculation is preferable (5).

This article introduces a web-based calculator designed to compute sample sizes for studies employing CFA and SEM. The web-based calculator is accessible at https://wnarifin.github.io/ssc_web.html web page under the "Structural Equation Modeling" heading. The calculator has been developed incrementally over the past five years starting from early 2020 based on the formulas and algorithms provided by Kim (4, 6), so I briefly describe important technical aspects of the calculator. I demonstrate sample size calculations with common examples for CFA and SEM using the web-based calculator modules. Critical comparisons with other available web-based calculators for SEM are also discussed.

© Malaysian Association of Education in Medicine and Health Sciences and Penerbit Universiti Sains Malaysia. Year
This work is licensed under the terms of the Creative Commons Attribution (CC BY) (http://creativecommons.org/licenses/by/4.0/).

DEVELOPMENT

The development of this calculator utilized the R programming language (7), the R Shiny package (8) and the OpenCPU API (9). It began with writing the sample size functions and their prerequisite functions. These functions were then compiled in an R script that is available at https://github.com/wnarifin/medicalstats-in-R GitHub page. Web pages providing different modules for the calculator were prepared using the R Shiny package and OpenCPU API, all of which rely on the R script

Sample Size Formulas

Kim (4) derived four sample size formulas for SEM based on the expected values of fit indices, which are comparative fit index (CFI, equation 6), root mean squared error of approximation (RMSEA, equation 7) McDonald's fit index (equation 8), Steiger's gamma (equation 9). The developed R script utilizes only the CFI and RMSEA formulas since these indices are commonly reported in SEM studies and recommended for reporting (1, 2). Degrees of freedom (df), non-centrality parameter (NCP) and model-implied correlation matrix are prerequisites for the sample size determination using the formulas for RMSEA and CFI. Therefore, these are described below. The R script is provided in Appendix 1; it is also available as ss_sem_fun.R in the functions folder on the provided GitHub page, with examples of using the functions in the ss_sem_examples.R script file.

Degrees of Freedom

The calculation of the degrees of freedom, *df*s for the proposed and baseline CFA models requires the number of items and factors. Given *p* number of items, the *df* for a proposed model is obtained as (1),

$$df = a - b$$

where *a* is the number of elements in the input variance-covariance matrix of the data, and *b* is the number of freely estimated parameters in the model. Freely estimated parameters are:

- Factor loadings, *FL* (excluding marker indicator variables as these are not freely estimated)
- Variances, *VAR* (for factors and errors)
- Covariances, *COVAR* (between factors and errors)
- Regressions, *REG*

Therefore, *a* and *b* are obtained as,

$$a = \frac{p(p+1)}{2},$$

$$b = FL + VAR + COV + REG$$

For a baseline model, when all relationships are fixed to 0 with only item variances freely estimated, *b* equals the number of items. These internal R functions for calculating the *df*s for proposed and baseline models are given in the R script (Appendix 1). Moreover, *df* can also be calculated based on specified lavaan (10) syntax as provided at https://wnarifin.shinyapps.io/ss_sem_df/.

Non-centrality Parameter

It is required to calculate the non-centrality parameter (NCP) value before using the methods in Kim (4). Kim provided the algorithm to obtain the NCP value in SPSS syntax and SAS programming language (4), given specified values of alpha, power and *df*. The algorithm was rewritten in R programming language and included in the R script (Appendix 1). The NCP value, mainly used as an internal function for the sample size calculator, is accessible via the calculator page at https://wnarifin.ocpu.io/sscalc/www/ssncp.html.

Model Correlation Matrix

Another prerequisite is the model-implied correlation matrix based on the estimated factor loading and factor correlation. This is required for calculating the sample size based on CFI. The model-implied correlation matrix is obtained as (1),

$$\Sigma = \Lambda_y \Psi \Lambda_y^{\top} + \Theta_{\epsilon},$$

where Σ is the $p \times p$ matrix for p item correlations; Λ_y is the $p \times m$ matrix of factor loadings with m factors; Ψ is the $m \times m$ matrix of factor correlations; and Θ_ε is the $p \times p$ the diagonal matrix of unique variances. The R functions to come up with the correlation matrix are included in the R script (Appendix 1) for equal and unequal numbers of items per factor.

Validation

The results from the R script were verified by comparing the outputs with the tables in Kim (4) by replicating the preset conditions in the paper. The R script used for the validation, including the NCP and sample size values, is included in Appendix 2. The outputs from the R functions matched the values given in Table 2 to 8 in Kim (4) for all the parameter values.

CONFIRMATORY FACTOR ANALYSIS

For calculating sample sizes in research involving CFA, three calculator modules are available:
1. Confirmatory factor analysis by RMSEA
2. Confirmatory factor analysis by CFI
3. Confirmatory factor analysis by CFI (advanced)

Confirmatory Factor Analysis by RMSEA

This module is accessible at https://wnarifin.shinyapps.io/ss_sem_rmsea/. The calculator allows the calculation of sample size for CFA based on the number of items and factors, given the expected RMSEA value. The default RMSEA value for the calculator is 0.05, which is the typical cutoff value for model fit using RMSEA. The interface is shown in Figure 1.

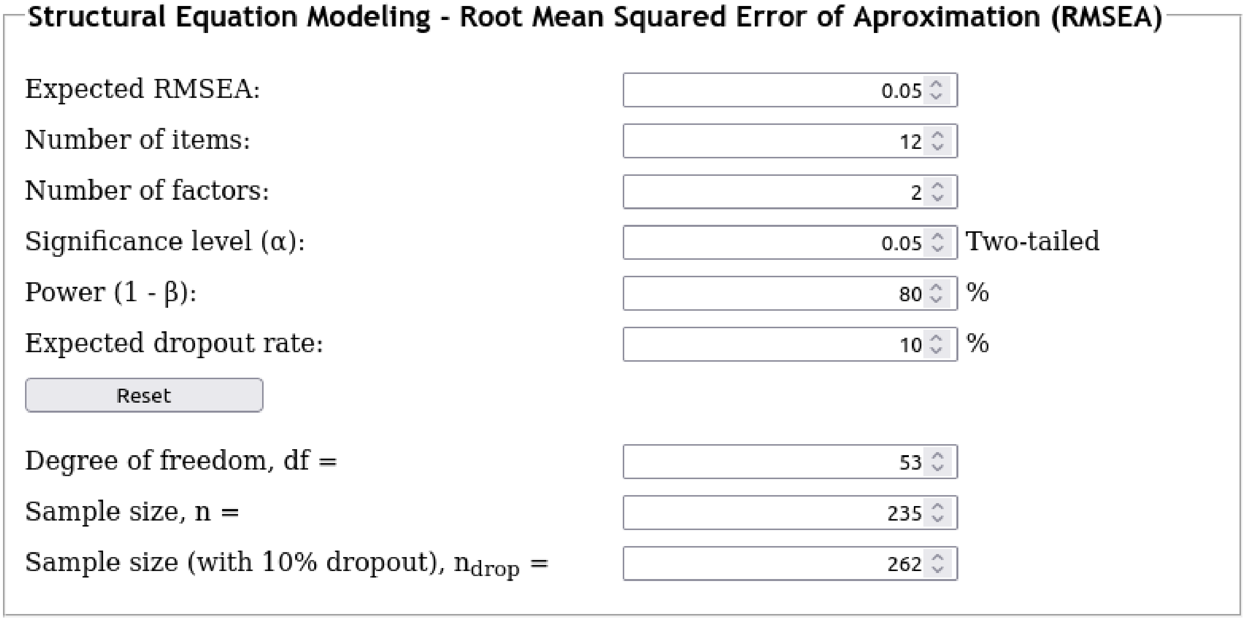

Figure 1. CFA by RMSEA module interface and example calculation

Let's say, a researcher wants to validate the ABC-Q questionnaire containing two factors. Factor 1 comprises eight items, and Factor 2 comprises four items. The acceptable RMSEA is 0.05 and below. A two-tailed significance level $\alpha = 0.05$ and a power of 80% are specified. The dropout rate is expected to be 10%. How many respondents should he sample?

The calculator provides the outputs below the "Reset" button (Figure 1). To verify the internal structure of ABC-Q, we need to sample 262 respondents, factoring in a 10% dropout rate. It additionally presents the computed *n* prior to considering the dropout rate and the model degrees of freedom.

Confirmatory Factor Analysis by CFI – For an Equal Number of Items Per Factor

This module is accessible at https://wnarifin.shinyapps.io/ss_sem_cfi_equal/. The calculator allows the calculation of sample size for CFA based on the expected CFI value, number of items, number of factors, average factor loading value, and average factor correlation value. The sample size calculation based on CFI requires more information as compared to the one based on RMSEA. However, it is important to note that since the module relies on generating a model-implied correlation matrix, this module should be used only **when each factor has an equal number of items**. For this, the calculator throws out an error message "Number of items must be multiples of factor!" when the number of items is not multiples of the factor. The default CFI value for the calculator is 0.95, which is the typical cutoff value for model fit using the fit index. The interface is shown in Figure 2.

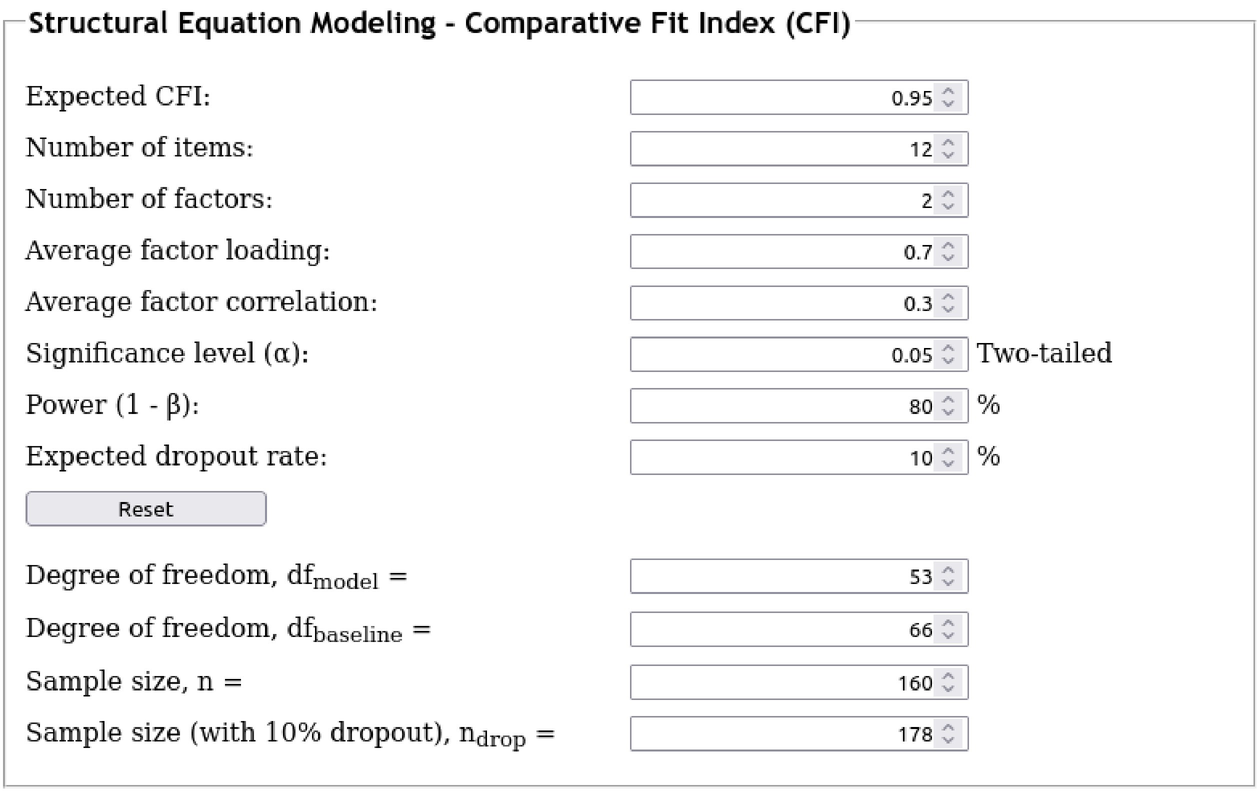

Figure 2. CFA by CFI module interface and example calculation.

Suppose a researcher wants to validate the ABC-Q questionnaire, which consists of two factors with six items in each factor. The researcher aims for a CFI of 0.95 and above. Based on previous studies, the average factor loading is around 0.7, and the average inter-factor correlation is approximately 0.3. The researcher specifies a two-tailed significance level α = 0.05 and a power of 80% are specified. The anticipated dropout rate is 10%. How many respondents are required for the study?

The calculator provides the outputs below the "Reset" button (Figure 2). To verify the internal structure validity of ABC-Q, we need to sample 178 respondents and account for a 10% dropout rate. The calculator also provides the calculated *n* before considering the dropout rate, and *df*s for the proposed and baseline models.

Confirmatory Factor Analysis by CFI (advanced) – for Unequal (and Equal) Number of Items Per Factor

This module is accessible at https://wnarifin.shinyapps.io/ss_sem_cfi_unequal/. The calculator allows the calculation of sample size for CFA based on the expected CFI value, number of items for each factor, average factor loading value, and average factor correlation value. In generating a model-implied correlation matrix, this module is more flexible as it allows calculating sample size when the factors have an equal or unequal number of items per factor. For this, the number of items for each factor is specified, separated by a comma, e.g. "4,3,2" for four, three and two items of three factors. The interface is shown in Figure 3.

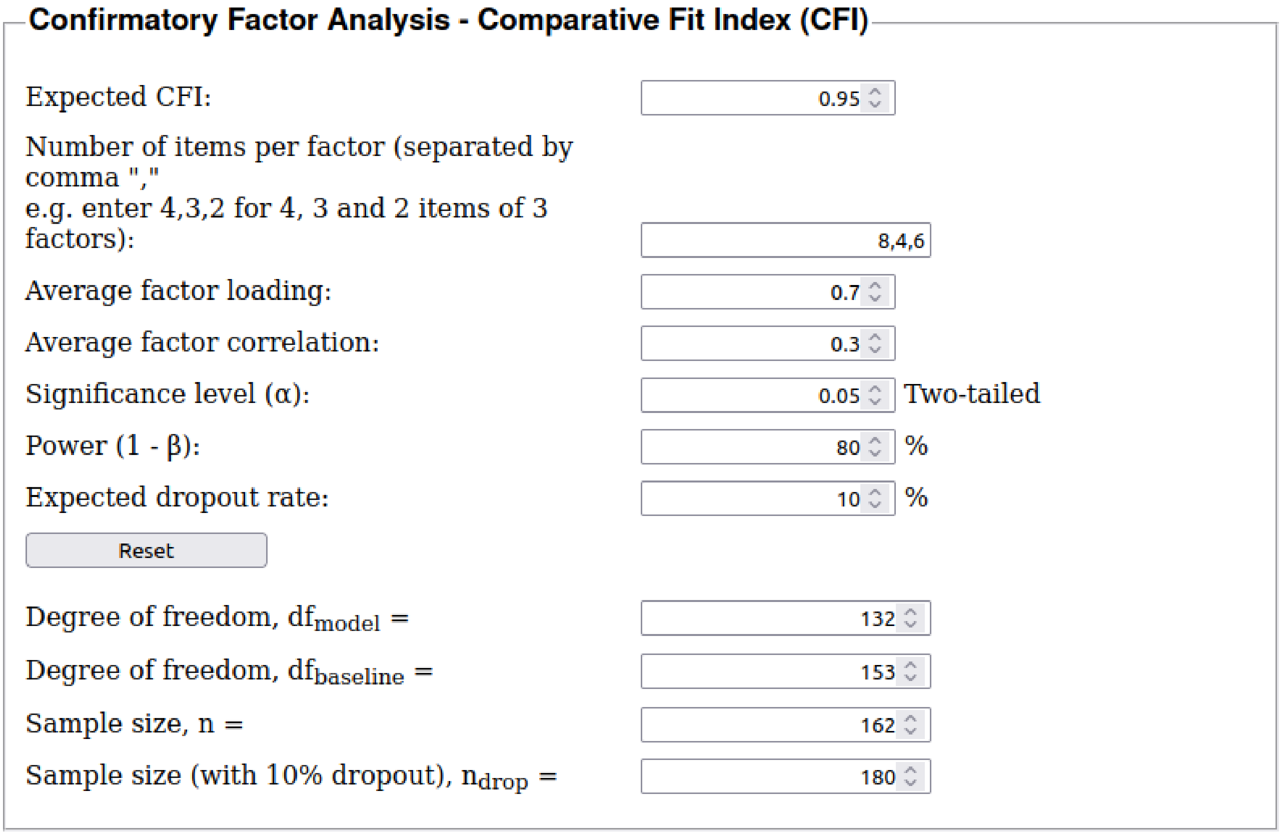

Figure 3. CFA by CFI (advanced) module interface and example calculation.

Consider a scenario where a researcher seeks to validate the ABC-Q questionnaire, which comprises three factors: Factor 1 containing eight items, Factor 2 consisting of four items, and Factor 3 including six items. The desired CFI is 0.95 or higher. According to previous studies, the average factor loading is approximately 0.7 and the average inter-factor correlation is around 0.3. A two-tailed significance level $\alpha = 0.05$ and a power of 80% are specified. A dropout rate of 10% is anticipated. How many respondents are required for this research?

The calculator provides the outputs below the "Reset" button (Figure 3). We must sample 180 respondents to confirm the internal structure validity of ABC-Q, taking into account a 10% dropout rate. It also provides the calculated *n* before considering the dropout rate, and *df*s for the proposed and baseline models.

STRUCTURAL EQUATION MODEL

Structural equation modeling by RMSEA (General)

To calculate the required sample sizes for studies involving SEM (which also includes CFA), the "Structural Equation Modeling by RMSEA (general)" calculator module can be used, accessible at https://wnarifin.ocpu.io/sscalc/www/ssrmsea.html. Currently, only the sample size calculation based on RMSEA is available. A web module of the sample size calculation for general SEM by CFI is planned for future development. At present, the CFI-based sample size determination can only be performed using the ncfi_calc() function in the provided R script. If the model *df* is not known, it can be calculated using the "Structural Equation Modeling – Degrees of Freedom" module for calculating the *df* at https://wnarifin.shinyapps.io/ss_sem_df/.

Suppose a researcher wants to validate the structural model given in Figure 4. The allowed RMSEA is 0.05 and below. A two-tailed significance level α = 0.05 and a power of 80% are specified. The dropout rate is expected to be 10%. How many respondents should he sample?

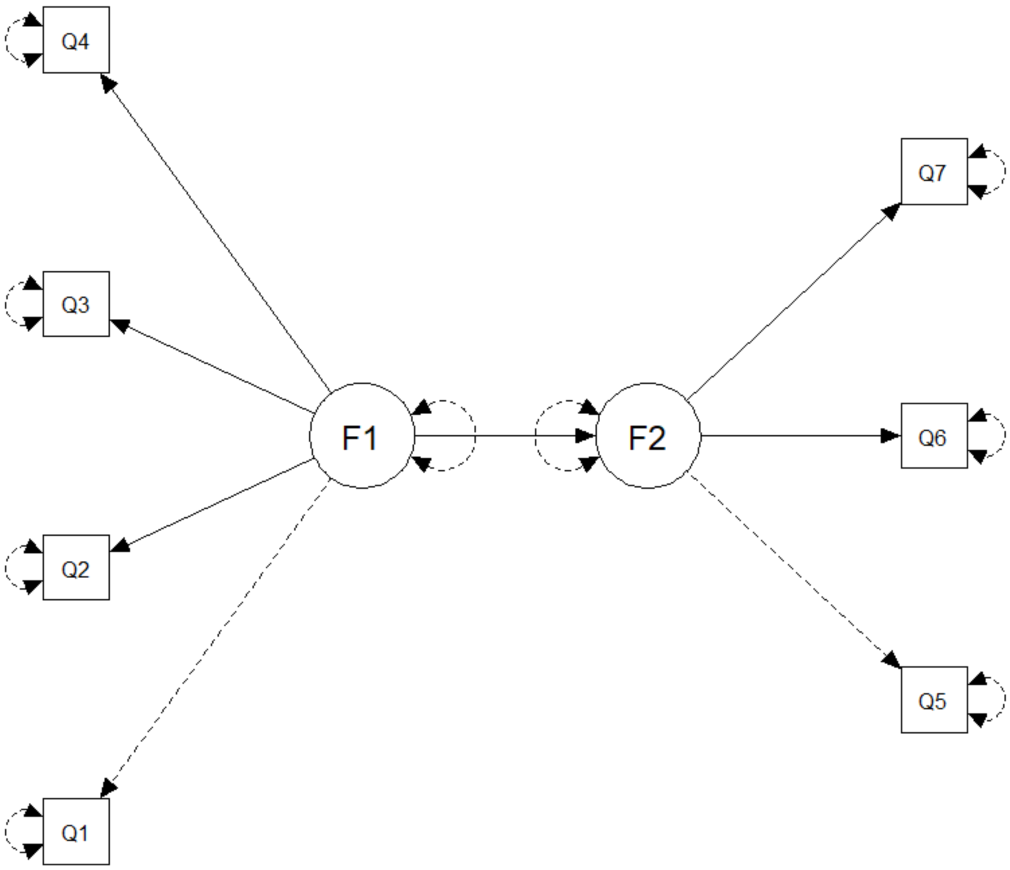

Figure 4. The proposed SEM model.

The sample size calculator module allows the calculation of sample size for SEM based on the proposed model's *df*, given the expected RMSEA value. The default RMSEA value for the calculator is 0.05, which is the typical cutoff value for model fit using RMSEA. As it requires *df*, we may start with obtaining the *df* for the proposed SEM model from the degrees of freedom calculator module (Figure 5) for the model in Figure 4. It requires specifying the model using *lavaan* syntax, which can be learned from https://lavaan.ugent.be/tutorial/syntax1.html.

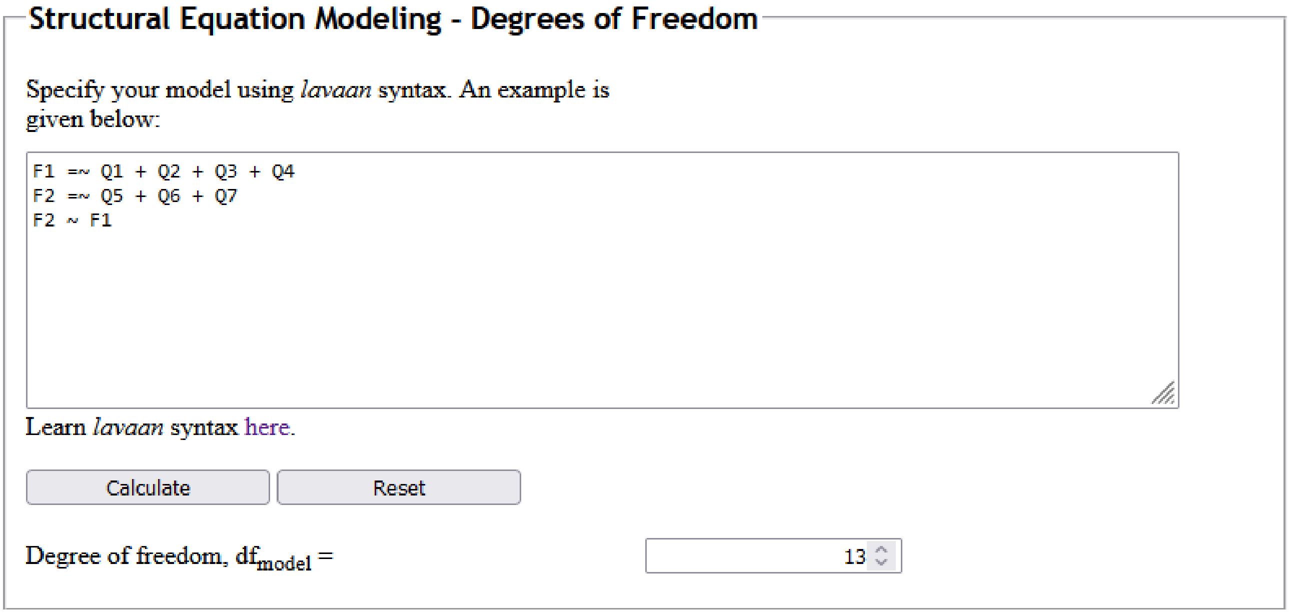

Figure 5. Degrees of freedom module interface and example calculation.

The calculator provides the *df* below the “Calculate” and “Reset” buttons (Figure 5). For the model, the model *df* is 13. Using this *df* value, open the sample size calculator module and enter this df and other relevant values. The calculator provides the outputs below the “Calculate” and “Reset” buttons (Figure 6).

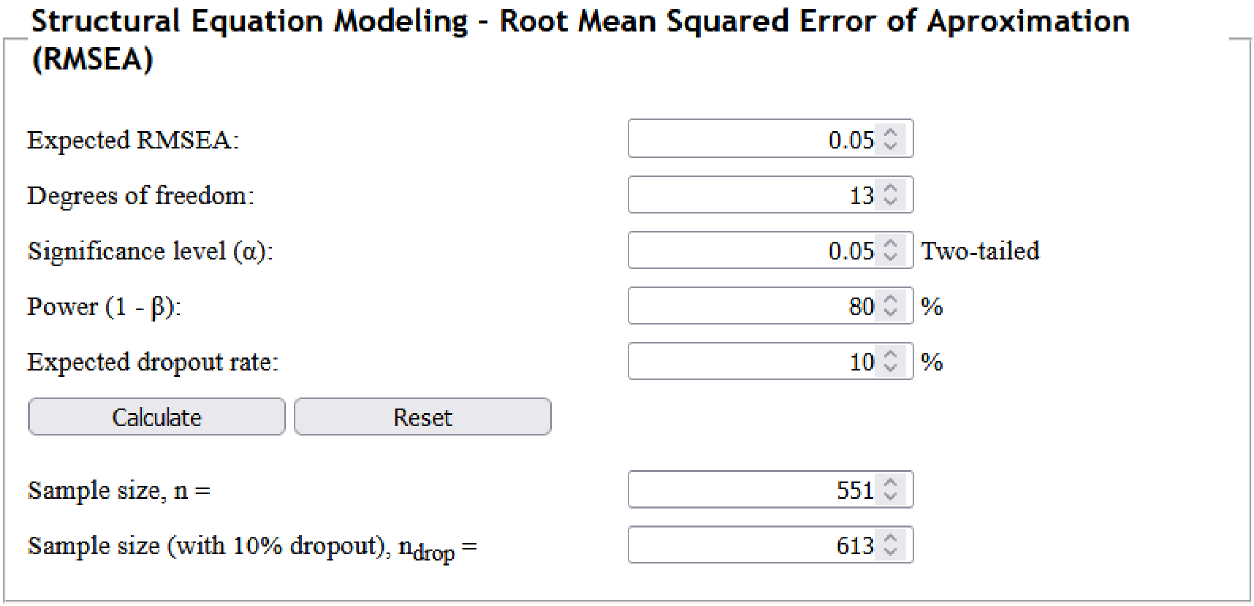

Figure 6. SEM by RMSEA module interface and example calculation.

The calculated *n* before considering the dropout rate is also provided. A sample of 551 respondents is required to assess the structural validity of the proposed SEM model, accounting for an anticipated 10% dropout rate.

DISCUSSION

The development of this web-based sample size calculator for SEM studies using the expected CFI and RMSEA values is described in this article. The functions underlying the R script that powers the calculator are based on the formulas and algorithm provided by Kim (4). The strength of the present calculator is that it requires easily obtained information for sample size determination for CFA (number of items and factors, average factor loading, average factor correlation) based on RMSEA and CFI. It also provides the sample size calculation for SEM based on RMSEA and the model *df* using the commonly used *lavaan* syntax.

Other than this calculator, other notable web-based sample size calculators for SEM mainly rely on the R programming language. Preacher and Coffman (11) (http://www.quantpsy.org/rmsea/rmsea.htm) and Gnambs (12) (https://timo.gnambs.at/research/power-for-sem) provided web-based R code generators for sample size determination for RMSEA, and a test of difference in RMSEA between nested model based on MacCallum, Browne and Sugawara (13) and MacCallum, Browne and Cai (14) respectively. Preacher and Coffman (11) allow users to submit the generated R code to an R web server for code execution. Gnambs (12) provides the R and SPSS code generator for determining sample size by Steiger's gamma and McDonald's fit indices based on the formulas in Kim (4). Gnambs (12) also provides a code generator for the goodness of fit index (GFI) and adjusted goodness of fit index (AGFI) based on the formulas in MacCallum and Hong (15). However, Gnambs's (12) implementation for RMSEA requires specifying H_0 and H_A, while the present calculator closely follows the implementation in Kim (4) which only requires specifying the target value of RMSEA. Both code generators require users to specify the *df* manually instead of the number of items and factors, while the present calculator does not require manual specification of the *df* for the sample size calculation for CFA.

Wang and Rhemtulla (16) developed *pwrSEM*, a web-based R Shiny application that allows estimation of the required sample size based on Monte Carlo simulation, which is available at https://yilinandrewang.shinyapps.io/pwrSEM/. However, the user must increase (or decrease) the sample size incrementally until an acceptable power is achieved. The user must also set parameter values for the specified model. Therefore, it can be difficult to use in practice if the user has no complete information about the values. Jobst, Bader and Moshagen (17) developed a web-based R Shiny interface (https://sempower.shinyapps.io/sempower/) for *semPower* R package that provides determination of sample size based on RMSEA, McDonald's, GFI and AGFI fit indices. However, users must specify the *df* manually in one of the menu options, while the present calculator provides more flexibility by allowing sample size determination by the number of items and factors for CFA, and *df* for SEM in general. Another notable mention is a web-based R Shiny application by Jak et al. (18) (https://sjak.shinyapps.io/power4SEM/) that allows the sample size determination based on NCP and RMSEA. This application also has the same issues with the previously mentioned calculators in terms of complexity and manual specification of the *df*.

This calculator provides sample size calculation for CFI, which is not available from any of the implementations described above. To my knowledge, this is not yet implemented in the form of a calculator elsewhere. This could be because a correlation matrix from the estimated factor loading and correlation values must be included in the calculation, specifically to obtain F_B in Kim's formula for CFI (4). Because of that, the R script that forms the basis for this web-based calculator also includes two functions to obtain the correlation matrix to facilitate the sample size determination for CFI.

CONCLUSION

In this article, a web-based calculator has been developed to assist in the determination of sample sizes for studies that use CFA and SEM. The calculator is accessible at https://wnarifin.github.io/ssc_web.html under the "Structural Equation Modeling" heading. The web-based calculator's modules were used to demonstrate sample size calculations for various CFA and SEM examples. This tool, which is accessible through any web browser, enables researchers to determine the required sample sizes for CFA and SEM studies based on commonly used fit indices, such as RMSEA and CFI. It is expected that this calculator will serve as a valuable resource for researchers in medical education and other scientific disciplines, assisting in research planning and the preparation of research proposal.

REFERENCES

1. Brown, TA. Confirmatory factor analysis for applied research. New York: The Guilford Press; 2015.
2. Kline R. Principles and practice of structural equation modeling. 4th ed. New York: The Guilford Press; 2016.
3. Muthen LK, Muthen BO. How to use a Monte Carlo study to decide on sample size and determine power. Structural Equation Modeling: A Multidisciplinary Journal. 2002;9(4), 599-620.
4. Kim KH. The relation among fit indexes, power, and sample size in structural equation modeling. Structural Equation Modeling. 2005;12(3):368-90.
5. Arifin WN. A web-based sample size calculator for reliability studies. Education in Medicine Journal. 2018;10(3):67–76. doi:10.21315/eimj2018.10.3.8
6. Arifin WN. A web-based sample size calculator for structural equation modeling in psychological research. Online oral session presented at: 3rd Biennial International Psychology Conference; 2021 October 25; Effat University, SA.
7. R Core Team. R: a language and environment for statistical computing. Vienna, Austria: R Foundation for Statistical Computing; 2024. Available from https://www.r-project.org/
8. Chang W, Cheng J, Allaire J, Sievert C, Schloerke B, Xie Y, et al. shiny: Web Application Framework for R. 2024. Available from https://shiny.posit.co/
9. Jeroen O. The OpenCPU System: Towards a Universal Interface for Scientific Computing through Separation of Concerns. arXiv:1406.4806 [stat.CO] 2014. Available from https://arxiv.org/abs/1406.4806
10. Rosseel Y. lavaan: An R Package for Structural Equation Modeling. *Journal of Statistical Software*. 2012;48(2):1-36. doi:10.18637/jss.v048.i02
11. Preacher KJ, Coffman DL. Computing power and minimum sample size for RMSEA [Computer software]. 2006 [cited 17 February 2020]. Available from http://quantpsy.org/rmsea/rmsea.htm
12. Gnambs T. Required sample size and power for SEM [Internet]. 2024 [cited 17 February 2020]. Available from https://timo.gnambs.at/research/power-for-sem
13. MacCallum RC, Browne MW, Sugawara, HM. Power analysis and determination of sample size for covariance structure modeling. Psychological Methods. 1996;1(2):130-149. doi:10.1037/1082-989X.1.2.130
14. MacCallum RC, Browne MW, Cai L. Testing differences between nested covariance structure models: Power analysis and null hypotheses. Psychological Methods. 2006;11(1):19-35. doi:10.1037/1082-989X.11.1.19
15. MacCallum RC, Hong S. Power Analysis in Covariance Structure Modeling Using GFI and AGFI. Multivariate Behavioral Research. 1997;32(2):193-210.

doi:10.1207/s15327906mbr3202_5

16. Wang YA, Rhemtulla M. Power analysis for parameter estimation in structural equation modeling: A discussion and tutorial. Advances in Methods and Practices in Psychological Science. 2021 Jan;4(1):2515245920918253.
17. Jobst LJ, Bader M, Moshagen M. A tutorial on assessing statistical power and determining sample size for structural equation models. Psychological Methods. 2023;28(1):207.
18. Jak S, Jorgensen TD, Verdam MG, Oort FJ, Elffers L. Analytical power calculations for structural equation modeling: A tutorial and Shiny app. Behavior Research Methods. 2021;53:1385-406.

APPENDIX

Appendix 1: R Script for Sample Size Calculation for SEM, **ss_sem_fun.R**

```
# Sample size calculator for SEM
# ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^

# Calculate model df
df_model = function(n_item, n_factor) {
  n_item = n_item
  n_factor = n_factor
  n_cor = n_factor*(n_factor - 1)/2
  b = n_item*(n_item + 1)/2
  a = (n_item - n_factor) + n_item + n_factor + n_cor  # freely estimated FL + error var + factor var + factor cor
  df = b - a  # model df
  return(df)
}

# Calculate baseline df
df_baseline = function(n_item, n_factor) {  # added back n_factor
  n_item = n_item
  n_factor = n_factor
  n_factor = n_cor = 0  # added for clarity, factor var, factor cor = 0
  b = n_item*(n_item + 1)/2
  a  = 0 + n_item + n_factor + n_cor  # FL, factor var, factor cor = 0, num error var = num indicator var
```

```
  dfb = b - a
  return(dfb)
}

# Calculate NCP given alpha, power and model df
ncp_calc = function(alpha, power, df) {
  crit = qchisq(1 - alpha, df)
  delta = round(crit - df)
  times = 1
  direc = 1
  amount = 10

  while (times < 9) {
    delta = delta + direc * amount
    pow = 1 - pchisq(crit, df, delta)
    if (direc * (power - pow) < 0) {
      times = times + 1
      direc = -1 * direc
      amount = amount / 10
    }
  }
  return(delta)
}

# Obtain model-implied correlation matrix, equal no of item per factor
cormat_equal = function(n_item, n_factor, fl, factor_cor) {
  n_item = n_item
  n_factor = n_factor
  fl  = fl
```

```
  cor_ = factor_cor
  if(n_item %% n_factor != 0){
    print("Number of items must be multiples of factor!")
  } else if(n_item %% n_factor == 0) {
    # FL per factor matrix
    mat_fl = matrix(rep(0, n_item*n_factor), ncol=n_factor)
    start_loc = 1
    n_per_factor = n_item/n_factor
    for(i in 1:n_factor) {
      end_loc = start_loc + n_per_factor - 1
      mat_fl[start_loc:end_loc, i] = rep(fl_ , n_per_factor)
      start_loc = end_loc + 1
    }
    # factor correlation matrix
    mat_fc = matrix(rep(cor_ , n_factor^2), ncol=n_factor)
    diag(mat_fc) = 1
    # unique variance matrix
    uvar = 1 - fl_^2
    uvar = diag(uvar, n_item, n_item)
    # correlation matrix
    mat_cor = mat_fl %*% mat_fc %*% t(mat_fl) + uvar
    return(mat_cor)
  }
}

# Obtain model-implied correlation matrix, unequal no of item per factor
cormat_unequal = function(vector_item, fl, factor_cor) {
  vec_item = vector_item
  n_item = sum(vec_item)
```

```
  n_factor = length(vec_item)
  fl_  = fl
  cor_  = factor_cor
  # FL per factor matrix
  mat_fl = matrix(rep(0, n_item*n_factor), ncol=n_factor)
  start_loc = 1
  for(i in 1:n_factor) {
    end_loc = start_loc + vec_item[i] - 1
    mat_fl[start_loc:end_loc, i] = rep(fl_ , vec_item[i])
    start_loc = end_loc + 1
  }
  # factor correlation matrix
  mat_fc = matrix(rep(cor_ , n_factor^2), ncol=n_factor)
  diag(mat_fc) = 1
  # unique variance matrix
  uvar = 1 - fl_^2
  uvar = diag(uvar, n_item, n_item)
  # correlation matrix
  mat_cor = mat_fl %*% mat_fc %*% t(mat_fl) + uvar
  return(mat_cor)
}

# Calculate sample size given expected RMSEA
nrmsea_calc = function(rmsea = 0.05, alpha, power, df) {
  ncp = ncp_calc(alpha, power, df)
  N_e = (ncp / (rmsea^2 * df)) + 1
  return(N_e)
}
```

```
# Calculate sample size given expected CFI
ncfi_calc = function(cfi = 0.95, alpha, power, df, dfb, cormat) {
  ncp = ncp_calc(alpha, power, df)
  F_B = -log(det(cormat))
  N_cfi = (ncp + dfb*(1 - cfi)) / (F_B*(1 - cfi)) + 1
  return(N_cfi)
}
```

Appendix 2: Code to reproduce the values in Kim (4).

Code to prepare the tables:

```
source("ss_sem_fun.R")

# NCP
df = c(1:30, 35, 40, 45, seq(50, 100, 10), seq(125, 250, 25), seq(300, 500, 50))
pow1 = .8
pow2 = .9

# Table 2
n1 = mapply(ncp_calc, alpha = 0.05, power = pow1, df = df)
n2 = mapply(ncp_calc, alpha = 0.05, power = pow2, df = df)
ncp = data.frame(df = df, n_power.80 = round(n1, 3), n_power.90 = round(n2, 3))

# CFI
cfi = c(.9, .95, .99, .9, .95, .99)
pow = c(.8, .8, .8, .9, .9, .9)

# Table 3, FL = .6
p = 6
n_fac = 2
fl = .6
f_cor = .3
```

```
df = 8  # df_model(p, n_fac)
dfb = df_baseline(p, n_fac)
cormat = cormat_equal(p, n_fac, fl, f_cor)

n1 = rep(0, length(cfi))
for (i in 1:length(cfi)) {
  n1[i] = ncfi_calc(cfi[i], 0.05, pow[i], df, dfb, cormat)
}

# Table 3, FL = .8
p = 6
n_fac = 2
fl = .8
f_cor = .3
df = 8  # df_model(p, n_fac)
dfb = df_baseline(p, n_fac)
cormat = cormat_equal(p, n_fac, fl, f_cor)

n2 = rep(0, length(cfi))
for (i in 1:length(cfi)) {
  n2[i] = ncfi_calc(cfi[i], 0.05, pow[i], df, dfb, cormat)
}

cfi_1 = data.frame(power = pow, cfi = cfi, n_cfiFL.6 = round(n1),
                   n_cfiFL.8 = round(n2))

# Table 4, FL = .6
p = 9
n_fac = 3
```

```
fl = .6
f_cor = .3
df = 24  # df_model(p, n_fac)
dfb = df_baseline(p, n_fac)
cormat = cormat_equal(p, n_fac, fl, f_cor)

n1 = rep(0, length(cfi))
for (i in 1:length(cfi)) {
  n1[i] = ncfi_calc(cfi[i], 0.05, pow[i], df, dfb, cormat)
}

# Table 4, FL = .8
p = 9
n_fac = 3
fl = .8
f_cor = .3
df = 24  # df_model(p, n_fac)
dfb = df_baseline(p, n_fac)
cormat = cormat_equal(p, n_fac, fl, f_cor)

n2 = rep(0, length(cfi))
for (i in 1:length(cfi)) {
  n2[i] = ncfi_calc(cfi[i], 0.05, pow[i], df, dfb, cormat)
}

cfi_2 = data.frame(power = pow, cfi = cfi, n_cfiFL.6 = round(n1), n_cfiFL.8
= round(n2))
cfi_2
```

```
# Table 5, FL = .6

p = 15

n_fac = 5

fl = .6

f_cor = .3

df = 80  # df_model(p, n_fac)

dfb = df_baseline(p, n_fac)

cormat = cormat_equal(p, n_fac, fl, f_cor)


n1 = rep(0, length(cfi))

for (i in 1:length(cfi)) {

  n1[i] = ncfi_calc(cfi[i], 0.05, pow[i], df, dfb, cormat)

}


# Table 5, FL = .8

p = 15

n_fac = 5

fl = .8

f_cor = .3

df = 80  # df_model(p, n_fac)

dfb = df_baseline(p, n_fac)

cormat = cormat_equal(p, n_fac, fl, f_cor)


n2 = rep(0, length(cfi))

for (i in 1:length(cfi)) {

  n2[i] = ncfi_calc(cfi[i], 0.05, pow[i], df, dfb, cormat)

}
```

```
cfi_3 = data.frame(power = pow, cfi = cfi, n_cfiFL.6 = round(n1), n_cfiFL.8
= round(n2))


# RMSEA

eps = c(.08, .05, .01, .08, .05, .01)

pow = c(.8, .8, .8, .9, .9, .9)


# Table 6

# p = 6

df = 8

n = mapply(nrmsea_calc, rmsea = eps, alpha = 0.05, power = pow, df = df)

rmsea_1 = data.frame(power = pow, rmsea = eps, n_rmsea = round(n))


# Table 7

# p = 9

df = 24

n = mapply(nrmsea_calc, rmsea = eps, alpha = 0.05, power = pow, df = df)

rmsea_2 = data.frame(power = pow, rmsea = eps, n_rmsea = round(n))


# Table 8

# p = 15

df = 80

n = mapply(nrmsea_calc, rmsea = eps, alpha = 0.05, power = pow, df = df)

rmsea_3 = data.frame(power = pow, rmsea = eps, n_rmsea = round(n))


# Reproduce Tables

# NCP

cbind(ncp[1:25,], ncp[26:50,])  # Table 2: NCP by df at alpha = 0.05

# CFI
```

```
cfi_1   # Table 3: n CFI for p = 6, df = 8, corr = .3
cfi_2   # Table 4: n CFI for p = 9, df = 24, corr = .3
cfi_3   # Table 5: n CFI for p = 15, df = 80, corr = .3
# RMSEA
rmsea_1   # Table 6: n for RMSEA for p = 6, df = 8
rmsea_2   # Table 7: n for RMSEA for p = 9, df = 24
rmsea_3   # Table 8: n for RMSEA for p = 15, df = 80
```

Outputs:

```
> # NCP
> cbind(ncp[1:25,], ncp[26:50,])  # Table 2: NCP by df at alpha = 0.05
   df n_power.80 n_power.90  df n_power.80 n_power.90
1   1      7.849     10.507  26     23.200     28.784
2   2      9.635     12.654  27     23.546     29.194
3   3     10.903     14.171  28     23.885     29.596
4   4     11.935     15.405  29     24.219     29.991
5   5     12.828     16.469  30     24.547     30.379
6   6     13.624     17.419  35     26.107     32.225
7   7     14.351     18.284  40     27.557     33.940
8   8     15.022     19.083  45     28.918     35.549
9   9     15.650     19.829  50     30.204     37.069
10 10     16.241     20.532  60     32.593     39.891
11 11     16.802     21.198  70     34.787     42.483
12 12     17.336     21.833  80     36.829     44.893
13 13     17.847     22.439  90     38.745     47.155
14 14     18.338     23.022 100     40.556     49.293
15 15     18.811     23.583 125     44.721     54.206
16 16     19.268     24.125 150     48.483     58.643
```

```
17 17      19.710      24.650 175      51.942      62.721
18 18      20.139      25.158 200      55.160      66.515
19 19      20.555      25.652 225      58.182      70.077
20 20      20.961      26.132 250      61.039      73.444
21 21      21.356      26.600 300      66.353      79.706
22 22      21.741      27.057 350      71.238      85.462
23 23      22.118      27.503 400      75.785      90.818
24 24      22.486      27.939 450      80.055      95.848
25 25      22.847      28.366 500      84.093     100.604
> # CFI
> cfi_1  # Table 3: n CFI for p = 6, df = 8, corr = .3
  power  cfi n_cfiFL.6 n_cfiFL.8
1   0.8 0.90       225        67
2   0.8 0.95       429       127
3   0.8 0.99      2061       607
4   0.9 0.90       280        83
5   0.9 0.95       539       159
6   0.9 0.99      2612       769
> cfi_2  # Table 4: n CFI for p = 9, df = 24, corr = .3
  power  cfi n_cfiFL.6 n_cfiFL.8
1   0.8 0.90       228        69
2   0.8 0.95       424       128
3   0.8 0.99      1990       597
4   0.9 0.90       276        83
5   0.9 0.95       519       156
6   0.9 0.99      2465       740
> cfi_3  # Table 5: n CFI for p = 15, df = 80, corr = .3
  power  cfi n_cfiFL.6 n_cfiFL.8
1   0.8 0.90       235        73
```

```
2   0.8 0.95      417      129
3   0.8 0.99     1872      578
4   0.9 0.90      275       85
5   0.9 0.95      496      154
6   0.9 0.99     2270      701
> # RMSEA
> rmsea_1  # Table 6: n for RMSEA for p = 6, df = 8
  power rmsea n_rmsea
1   0.8  0.08     294
2   0.8  0.05     752
3   0.8  0.01   18779
4   0.9  0.08     374
5   0.9  0.05     955
6   0.9  0.01   23854
> rmsea_2  # Table 7: n for RMSEA for p = 9, df = 24
  power rmsea n_rmsea
1   0.8  0.08     147
2   0.8  0.05     376
3   0.8  0.01    9370
4   0.9  0.08     183
5   0.9  0.05     467
6   0.9  0.01   11642
> rmsea_3  # Table 8: n for RMSEA for p = 15, df = 80
  power rmsea n_rmsea
1   0.8  0.08      73
2   0.8  0.05     185
3   0.8  0.01    4605
4   0.9  0.08      89
5   0.9  0.05     225
```

```
6   0.9  0.01    5613
```