A sensitivity analysis of predicting continued use of online teacher professional development and the influence of social presence and sociability by Jo A. Smith and Stephen A. Sivo

Gail Fish (University of Florida) & YongSeok Lee (University of Florida)

2021-10-12

#####Original Paper: Smith,J.A.,& Sivo,S.A.(2012). Predicting continued use of online teacher professional development and the influence of social presence and sociability. *British Journal of Educational Technology, 43*(6), 871-882. doi:10.1111/j.1467-8535.2011.01223.x

SENSITIVITY ANALYSIS USING SEMsens

This document provides a simple example analysis of a path analysis dataset, a survey of teachers enrolled in a statewide online reading course. The study examines how a Technology Acceptance Model (TAM) could predict teachers' intentions to continue using e-learning for professional development based on perceived ease of use and usefulness, as well as examine mediating influences of social presence and sociability in e-learning professional development.

The dataset has six manifest variables: Perceived Usefulness (PU), Perceived Ease of Use (PEU), Teachers' Reading Knowledge Assessment gains (Gains), Social Presence (SP), Sociability (SOC), and Continuance Intention (CI).

In addition to the `SEMsens` package, this vignette also makes use of `lavaan`.

```
#Load the packages
require(SEMsens)
require(lavaan)
set.seed(1)
```

Step 1: Original Path Model & Estimation

Here, we reproduce the the correlation matrix found in the article. First we create the lower diagonal and then convert to a covariance matrix and label the variables with `getCov()` from `lavaan`.

```
#Set a correlation matrix
lower = '
1.00
0.68 1.00
0.54 0.55 1.00
0.65 0.63 0.67 1.00
0.33 0.37 0.68 0.54 1.00
-0.01 0.00 0.03 -0.04 0.07 1.00'
```

```
#convert to full covariance matrix, using function from lavaan
full = getCov(lower, sds= c(4.61,5.37,7.25,3.44,8.91,8.80),
              names = c("PU", "PEU", "SP", "CI","SOC","Gains"))
```

We next set up the path model from the article, using `lavaan` model syntax with `sem` function. Through this code, we can get the result of (standardized) path coefficients and model fit indices. Standardized coefficient and model fit of this test almost exactly reproduces the results of the original paper (Smith & Sivo,2012). Slight differences are a result of using different statistical software (R or LISREL).

```
# Original model
lav_model <-  'SP~SOC
Gains~SP
PU~SP+PEU
PEU~SP
CI~SP+PU+PEU+SOC
Gains ~~ 0*CI
'
# Fit the original model with sem function
modelFit <-  sem(lav_model, sample.nobs=517, sample.cov=full, fixed.x=TRUE, std.lv=TRUE)
summary(modelFit, standardized = TRUE) #look at Std.all
```

```
## lavaan 0.6-9 ended normally after 24 iterations
##
##   Estimator                                         ML
##   Optimization method                           NLMINB
##   Number of model parameters                        14
##
##   Number of observations                           517
##
## Model Test User Model:
##
##   Test statistic                                 9.567
##   Degrees of freedom                                 6
##   P-value (Chi-square)                           0.144
##
## Parameter Estimates:
##
##   Standard errors                             Standard
##   Information                                 Expected
##   Information saturated (h1) model          Structured
##
## Regressions:
##                    Estimate  Std.Err  z-value  P(>|z|)   Std.lv  Std.all
##   SP ~
##     SOC               0.553    0.026   21.087    0.000    0.553    0.680
##   Gains ~
##     SP                0.036    0.053    0.682    0.495    0.036    0.030
##   PU ~
##     SP                0.151    0.024    6.404    0.000    0.151    0.238
##     PEU               0.471    0.032   14.775    0.000    0.471    0.549
##   PEU ~
##     SP                0.407    0.027   14.974    0.000    0.407    0.550
##   CI ~
##     SP                0.128    0.020    6.283    0.000    0.128    0.268
##     PU                0.226    0.029    7.746    0.000    0.226    0.302
##     PEU               0.134    0.025    5.318    0.000    0.134    0.209
##     SOC               0.069    0.015    4.769    0.000    0.069    0.179
##
## Covariances:
##                    Estimate  Std.Err  z-value  P(>|z|)   Std.lv  Std.all
##  .Gains ~~
##    .CI                0.000                               0.000    0.000
##
## Variances:
##                    Estimate  Std.Err  z-value  P(>|z|)   Std.lv  Std.all
##    .SP               28.203    1.754   16.078    0.000   28.203    0.538
##    .Gains            77.221    4.803   16.078    0.000   77.221    0.999
##    .PU               10.565    0.657   16.078    0.000   10.565    0.498
##    .PEU              20.075    1.249   16.078    0.000   20.075    0.697
##    .CI                4.651    0.289   16.078    0.000    4.651    0.392
```

```
fitMeasures(modelFit)
```

```
##                npar                fmin               chisq                  df
##              14.000               0.009               9.567               6.000
##              pvalue      baseline.chisq         baseline.df     baseline.pvalue
##               0.144            1359.242              15.000               0.000
##                 cfi                 tli                nnfi                 rfi
##               0.997               0.993               0.993               0.982
##                 nfi                pnfi                 ifi                 rni
##               0.993               0.397               0.997               0.997
##                logl   unrestricted.logl                 aic                 bic
##           -7436.961           -7432.177           14901.922           14961.394
##              ntotal                bic2               rmsea      rmsea.ci.lower
##             517.000           14916.955               0.034               0.000
##      rmsea.ci.upper        rmsea.pvalue                 rmr          rmr_nomean
##               0.072               0.711               1.038               1.038
##                srmr        srmr_bentler srmr_bentler_nomean                crmr
##               0.020               0.020               0.020               0.024
##         crmr_nomean          srmr_mplus   srmr_mplus_nomean               cn_05
##               0.024               0.020               0.020             681.482
##               cn_01                 gfi                agfi                pgfi
##             909.558               0.994               0.978               0.284
##                 mfi                ecvi
##               0.997               0.073
```

We can get same results by using `lavannify`,`lavaan` and the `standardizedsolution` functions. These are all in the `lavaan` package and present more focused results for standardized path coefficients and their standard error and p-values. Depending on users' research questions, it is possible to select results for individual pathways in the model.

```
smith_original <- lavaan::lavaanify(model = lav_model, auto = TRUE, model.type = "sem", fixed.x = TRUE)
smith_original <- lavaan::lavaan(model = smith_original, sample.cov = full, sample.nobs = 517)
smith_original_par <- lavaan::standardizedSolution(smith_original, type = "std.all")
smith_original_par #4th row and 7th column of table : smith_original_par[1:4,1:7]
```

```
##      lhs op   rhs est.std    se       z pvalue ci.lower ci.upper
## 1     SP  ~   SOC   0.680 0.021  32.801  0.000    0.639    0.721
## 2  Gains  ~    SP   0.030 0.044   0.683  0.495   -0.056    0.116
## 3     PU  ~    SP   0.238 0.037   6.492  0.000    0.166    0.310
## 4     PU  ~   PEU   0.549 0.033  16.453  0.000    0.484    0.615
## 5    PEU  ~    SP   0.550 0.030  18.226  0.000    0.491    0.609
## 6     CI  ~    SP   0.268 0.042   6.343  0.000    0.186    0.351
## 7     CI  ~    PU   0.302 0.038   7.852  0.000    0.227    0.378
## 8     CI  ~   PEU   0.209 0.039   5.347  0.000    0.132    0.286
## 9     CI  ~   SOC   0.179 0.037   4.804  0.000    0.106    0.252
## 10 Gains ~~    CI   0.000 0.000      NA     NA    0.000    0.000
## 11    SP ~~    SP   0.538 0.028  19.068  0.000    0.482    0.593
## 12 Gains ~~ Gains   0.999 0.003 378.978  0.000    0.994    1.004
## 13    PU ~~    PU   0.498 0.031  16.195  0.000    0.438    0.558
## 14   PEU ~~   PEU   0.697 0.033  21.013  0.000    0.632    0.763
## 15    CI ~~    CI   0.392 0.026  15.201  0.000    0.342    0.443
## 16   SOC ~~   SOC   1.000 0.000      NA     NA    1.000    1.000
```

Step 2: Construct the Sensitivity Model

After checking the original path model, we then create the sensitivity model using a **Phantom Variable**. A phantom variable is modeled with paths to all other variables to see the trajectory of estimates in the original model affected by specification of the Phantom variable. As shown in the code below, the phantom variable follows the normal distribution which has mean of zero and variance of one.

```
# Sensitivity model, with sensitivity parameters for all variables
sens_model <-  'SP~SOC
    Gains ~ SP
    PU ~ SP+PEU
    PEU ~ SP
    CI ~ SP+PU+PEU+SOC
    Gains ~~ 0*CI
    SP ~ phantom1*phantom
    Gains ~ phantom2*phantom
    PU ~ phantom3*phantom
    PEU ~ phantom4*phantom
    CI ~ phantom5*phantom
    SOC ~ phantom6*phantom
    phantom =~ 0  #mean of zero
    phantom ~~ 1*phantom  # variance of one'
```

Step 3: Conducting Sensitivity Analysis

Based on the specified `sens_model`, we can run the sensitivity analysis through `sa.aco()` function in `SEMsens` package. Note that we run with the parameters `k = 5` and `max.iter = 20` for a simple illustration. The default values for these parameters are `k = 50` and `max.iter = 1000`. For the other options, see the paper or vignette of `SEMsens` package (https://cran.r-project.org/web/packages/SEMsens/index.html).

```
smith_example <- sa.aco(
  sample.cov = full,
  sample.nobs = 517,
  model = lav_model,
  sens.model = sens_model,
  opt.fun = 1,
  paths = c(1:9),
  max.iter = 20,
  k = 5)
```

```
## Number of tried evaluations is 1.
## Number of converged evaluations is 1.
## Number of tried evaluations is 2.
## Number of converged evaluations is 2.
## Number of tried evaluations is 3.
## Number of converged evaluations is 3.
## Number of tried evaluations is 4.
## Number of converged evaluations is 4.
## Number of tried evaluations is 5.
## Number of converged evaluations is 5.
```

Step 4: Sensitivity Analysis Results

We can get the sensitivity analysis results after 5 iterations. The **sens.tables** function helps us to summarize of sensitivity analysis. In the smith_tables results, the **sens.summary** table contains estimates and p-values for each path in the original model information suggested in Step 1. It also provides the minimum, mean and maximum path estimates during sensitivity analysis.

```
smith_tables <- sens.tables(smith_example)
smith_tables$sens.summary
```

```
##            model.est model.pvalue mean.est.sens min.est.sens max.est.sens
## Gains~SP 0.03000005 4.947475e-01    0.03078806   0.02735571   0.03562662
## CI~SOC   0.17912707 1.553400e-06    0.17525972   0.14255763   0.20699081
## CI~PEU   0.20913951 8.949231e-08    0.20793924   0.16071079   0.24031659
## PU~SP    0.23799284 8.462697e-11    0.23152339   0.22321878   0.24179787
## CI~SP    0.26848140 2.255842e-10    0.27010031   0.22550080   0.32256848
## CI~PU    0.30228504 3.996803e-15    0.31350603   0.26999750   0.39231625
## PEU~SP   0.55000002 0.000000e+00    0.54972379   0.54023885   0.56146020
## PU~PEU   0.54910394 0.000000e+00    0.55356443   0.53608380   0.57792956
## SP~SOC   0.68000001 0.000000e+00    0.68091945   0.67523652   0.68849064
```

The result of **phan.paths** suggests the minimum, mean and maximum value of sensitivity parameters which were formed in the relationship between phantom variable and each variables in the path model during the iteration of Ant Colony Optimization (ACO).

```
smith_tables$phan.paths
```

```
##                  mean.phan    min.phan  max.phan
## PEU~phantom   -0.032628286 -0.16322067 0.1521077
## SOC~phantom    0.001431526 -0.08418588 0.1103978
## SP~phantom     0.002450675 -0.06472979 0.1227683
## Gains~phantom  0.003651090 -0.02906461 0.0630220
## CI~phantom     0.017121906 -0.17295276 0.2147725
## PU~phantom     0.036459926 -0.21132376 0.1887766
```

The table of **phan.min** indicates the sensitivity parameters for each path that led to smallest size of path estimates during the iteration process of ACO.

```
smith_tables$phan.min
```

```
##           SP~phantom Gains~phantom  PU~phantom   PEU~phantom  CI~phantom
## SP~SOC    0.05161516   -0.02637793  0.11714527 -0.0008575855  0.12632124
## Gains~SP  0.12276830    0.03659312  0.05607790 -0.1632206676 -0.17060386
## PU~SP     0.05161516   -0.02637793  0.11714527 -0.0008575855  0.12632124
## PU~PEU   -0.06426463   -0.02591714 -0.21132376 -0.0438425624  0.21477253
## PEU~SP   -0.03313566    0.06302200  0.18877664 -0.1073283518  0.08807237
## CI~SP    -0.06472979   -0.02906461  0.03162358  0.1521077372 -0.17295276
## CI~PU    -0.03313566    0.06302200  0.18877664 -0.1073283518  0.08807237
## CI~PEU    0.12276830    0.03659312  0.05607790 -0.1632206676 -0.17060386
## CI~SOC    0.12276830    0.03659312  0.05607790 -0.1632206676 -0.17060386
##          SOC~phantom
## SP~SOC    0.11039776
## Gains~SP -0.07264936
## PU~SP     0.11039776
## PU~PEU   -0.03585607
## PEU~SP   -0.08418588
## CI~SP     0.08945118
## CI~PU    -0.08418588
## CI~PEU   -0.07264936
## CI~SOC   -0.07264936
```

Similar to phan.min case, **phan.max** table provides the sensitivity parameters for each path that resulted in the largest size of path estimates during the process of ACO.

```
smith_tables$phan.max
```

```
##           SP~phantom Gains~phantom  PU~phantom PEU~phantom  CI~phantom
## SP~SOC    0.12276830    0.03659312  0.05607790 -0.16322067 -0.17060386
## Gains~SP -0.03313566    0.06302200  0.18877664 -0.10732835  0.08807237
## PU~SP    -0.06472979   -0.02906461  0.03162358  0.15210774 -0.17295276
## PU~PEU   -0.03313566    0.06302200  0.18877664 -0.10732835  0.08807237
## PEU~SP    0.12276830    0.03659312  0.05607790 -0.16322067 -0.17060386
## CI~SP     0.12276830    0.03659312  0.05607790 -0.16322067 -0.17060386
## CI~PU    -0.06426463   -0.02591714 -0.21132376 -0.04384256  0.21477253
## CI~PEU   -0.03313566    0.06302200  0.18877664 -0.10732835  0.08807237
## CI~SOC   -0.06472979   -0.02906461  0.03162358  0.15210774 -0.17295276
##          SOC~phantom
## SP~SOC   -0.07264936
## Gains~SP -0.08418588
## PU~SP     0.08945118
## PU~PEU   -0.08418588
## PEU~SP   -0.07264936
## CI~SP    -0.07264936
## CI~PU    -0.03585607
## CI~PEU   -0.08418588
## CI~SOC    0.08945118
```

The final **p.paths** table covers not only the p-values of original model's path estimates at the first column (default significance level: 0.05) but the final p-value of each path estimates that reverse the null-hypothesis decision of original path estimates. From the third column of table, sensitivity parameters are suggested that leads to the change of p-value. An **NA** result in the table occurs if there is no change in p-value and meaningful sensitivity parameters that changed p-value in the `sa.aco` function.

```
smith_tables$p.paths
```

```
##               p.value p.changed SP~phantom Gains~phantom PU~phantom PEU~phantom
## SP~SOC   0.000000e+00        NA         NA            NA         NA          NA
## Gains~SP 4.947475e-01        NA         NA            NA         NA          NA
## PU~SP    8.462697e-11        NA         NA            NA         NA          NA
## PU~PEU   0.000000e+00        NA         NA            NA         NA          NA
## PEU~SP   0.000000e+00        NA         NA            NA         NA          NA
## CI~SP    2.255842e-10        NA         NA            NA         NA          NA
## CI~PU    3.996803e-15        NA         NA            NA         NA          NA
## CI~PEU   8.949231e-08        NA         NA            NA         NA          NA
## CI~SOC   1.553400e-06        NA         NA            NA         NA          NA
##          CI~phantom SOC~phantom
## SP~SOC           NA          NA
## Gains~SP         NA          NA
## PU~SP            NA          NA
## PU~PEU           NA          NA
## PEU~SP           NA          NA
## CI~SP            NA          NA
## CI~PU            NA          NA
## CI~PEU           NA          NA
## CI~SOC           NA          NA
```

References

Leite, W., Shen, Z., Marcoulides, K., Fish, C., & Harring, J. (in press). Using ant colony optimization for sensitivity analysis in structural equation modeling. Structural Equation Modeling: A Multidisciplinary Journal.