A more efficient and consistent way of fitting PLS structural models: A better alternative to SEM than traditional PLS.

As published in Benchmarks Research Matters, February 2017

https://benchmarks.it.unt.edu/researchmatters

Jon Starkweather, PhD

Jon Starkweather, PhD jonathan.starkweather@unt.edu Consultant Research and Statistical Support



A green light to greatness."

https://www.unt.edu



http://it.unt.edu/research

R&SS hosts a number of "Short Courses". A list of them is available at: http://it.unt.edu/researchshortcourses

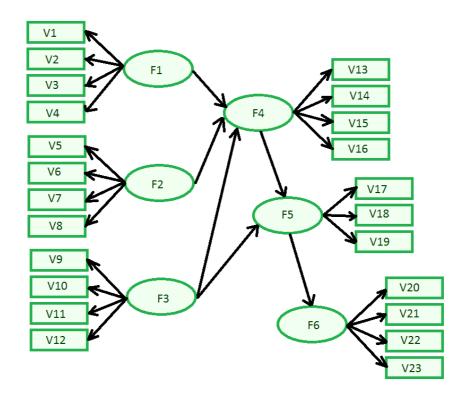
Those interested in learning more about R, or how to use it, can find information here: http://bayes.acs.unt.edu:8083/BayesContent/class/Jon/R_SC/

A more efficient and consistent way of fitting PLS structural models: A better alternative to SEM than traditional PLS.

Partial Least Squares (PLS) modeling (Wold, 1965, 1966a, 1973) or 'soft modeling' (Wold, 1982; Faulk, & Miller, 1992) was discussed in this column several years back (Starkweather, 2011). PLS is an important alternative to traditional path modeling and / or structural equation modeling (SEM) when the data in hand does not conform to the assumptions of those modeling techniques. However, PLS modeling does have its drawbacks. Early on Dijkstra (1983) revealed a lack of consistency when PLS is used to estimate structural models. Other researchers (Wold, 1982; Fornell & Bookstein, 1982) have noted that "PLS does not solve a global optimization problem for parameter estimation, indicating that there exists no single criterion consistently minimized or maximized to determine model parameter estimates" (Hwang & Takane, 2004, p. 1). Hwang and Takane also point out that PLS offers no global goodness of fit statistic which would allow model comparisons. In response to the criticisms above, several researchers have proposed alternative methods for, or modifications to, the traditional PLS approach. As stated in the 'matrixpls' package vignette (Ronkko, 2016c):

"Hwang and Takane (2014; 2004) proposed generalized structured component analysis (GSCA) arguing that it is superior over PLS because it has an explicit optimization criterion, which the PLS algorithm lacks. Dijkstra (2011; Dijkstra and Henseler 2015b; Dijkstra and Henseler 2015a) proposed that PLS can be made consistent by applying disattenuation, referring to this estimator as PLSc. Huang (2013; Bentler and Huang 2014) proposed two additional estimators that parameterize LISREL estimators based on Dijkstra's PLSc estimator. These estimators, referred to as PLSe1 an PLSe2 are argued to be more efficient than the consistent PLSc estimator" (p. 2).

Naturally, the focus of the current article is the 'matrixpls' package (Ronkko, 2016a) and its capabilities to use the new methods mentioned in the preceding paragraph. Below a simulated dataset is used to demonstrate the 'matrixpls' function of the 'matrixpls' package fitting the following model.



First, import the simulated data from the Research and Statistical Support (R&SS) server¹ and load the 'matrixpls' package (Ronkko, 2016a).

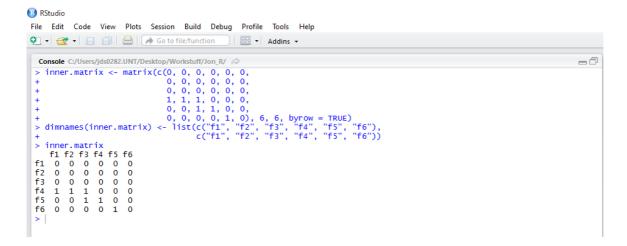
RStudio

```
Plie Edit Code View Plots Session Build Debug Profile Tools Help
Plie Edit Code View Plots Session Build Debug Profile Tools Help
Plie Edit Code View Plots Session Build Debug Profile Tools Help
Plie Edit Code View Plots Session Build Debug Profile Tools Help
Console C/Users/ids0282.UNT/Desktop/Workstuff/Jon_R/ 
R version 3.3.2 (2016-10-31) -- "Sincere Pumpkin Patch"
Copyright (C) 2016 The R Foundation for Statistical Computing
Platform: x86_64-w64-mingw32/x64 (64-bit)
R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'lenseD' or 'licenceD' for distribution details.
R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.
Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.
> pls.df <- read.csv("http://bayes.acs.unt.edu:8083/BayesContent/class/Jon/ExampleData/matPLS.csv")
> library(matrixpls)'
Attaching package: 'matrixpls'
The following objects are masked from 'package:stats':
    ave, loadings
> |
```

Next, we specify the structural model using the 3 matrices style (i.e. inner, outer or reflective, and formative). We begin by creating the inner matrix (i.e. a matrix specifying the unobserved variable relationships). Keep in mind, the 'matrixpls' function can recognize multiple methods for specifying a model (e.g., using 'lavaan' package syntax; Yves, 2012a & 2012b, as well as using the 'semPLS'

¹http://it.unt.edu/research

package syntax; Monecke, & Leisch, 2012); the example below shows how the inner matrix would be specified for the 'plspm' package (Sanchez, Trinchera, & Russolillo, 2016) and function, which is also accepted by the 'matrixpls' function.



Next, we create the outer matrix (i.e. specify the relationships between observed variables and the unobserved variables). Later, this matrix will be referred to as the 'reflective' matrix because it specifies those types of relationships.

File Edit Code View Plots Session Build Debug Profile Tools Help 💽 + 🕣 + 🕞 😥 🚔 🌈 Go to file/function 🔤 🔯 + Addins +	
<pre>Console C:/Users/ids0282.UNT/Desktop/Workstuff/Jon_K/ ↔ > outer.matrix <- matrix(rep(0, 23*6), ncol = 6) > outer.matrix[1:4, 1] <- 1 > outer.matrix[1:5, 2] <- 1 > outer.matrix[1:1:6, 3] <- 1 > outer.matrix[1:1:6, 4] <- 1 > outer.matrix[2:1:1:9, 5] <- 1 > outer.matrix[2:1:1:9, 5] <- 1 > outer.matrix[2:1:1:9, 5] <- 1 > outer.matrix f1 f2 f3 f4 f5 f6 V1 1 0 0 0 0 0 0 V2 1 0 0 0 0 0 0 V3 1 0 0 0 0 0 0 V4 1 0 0 0 0 0 0 V4 1 0 0 0 0 0 V5 0 1 1 0 0 0 0 V5 0 1 1 0 0 0 0 V6 0 1 1 0 0 0 0 V7 0 1 0 0 0 0 V10 0 0 1 0 0 0 V10 0 0 1 0 0 0 V11 0 0 1 0 0 0 V12 0 0 1 0 0 0 V13 0 0 0 1 0 0 V14 0 0 0 1 0 0 V15 0 0 0 1 0 0 V15 0 0 0 1 0 0 V16 0 0 1 0 0 V17 0 0 0 0 1 0 V17 0 0 0 0 1 0 V18 0 0 0 1 0 0 V19 0 0 0 1 0 V19 0 0 0 0 1 0 V10 0 0 1 0 V10 0 0 1 0 V11 0 0 0 0 0 V10 0 0 1 0 V11 0 0 0 0 0 V12 0 0 0 0 0 1 V12 0 0 0 0 0 1 V13 0 0 0 0 1 0 V14 0 0 0 1 0 V15 0 0 0 1 0 V15 0 0 0 1 0 V16 0 0 0 1 0 V17 0 0 0 0 1 0 V17 0 0 0 0 1 0 V18 0 0 0 0 1 0 V19 0 0 0 0 0 1 V21 0 0 0 0 0 1 V22 0 0 0 0 0 0 1 V22 0 0 0 0 0 0 1 V23 0 0 0 0 0 0 0 0 1 V23 0 0 0 0 0 0 0 0 1 V23 0 0 0 0 0 0 0 0 1 V23 0 0 0 0 0 0 0 0 0 0 V23 0 0 0 0 0 0 0 0 0 V23 0 0 0 0 0 0 0 0 0 V23 0 0 0 0 0 0 0 0 0 V20 0 0 0 0 0 0 0 0 0 V20 0 0 0 0 0 0 0 0 0 0 V20 0 0 0 0 0 0 0 0 0 V20 0 0 0 0 0 0 0 0 V20 0 0 0 0 0 0 0 0 0 0 0 V20 0 0 0 0 0 0 0 0 0 V20 0 0 0 0 0 0 0 0 0 V20 0 0 0 0 0 0 0 0 0 V20 0 0 0 0 0 0 0 0 0 0 V20 0 0 0 0 0 0 0 0 0 V20 0 0 0 0 0 0 0 0 0 V20 0 0 0 0 0 0 0 0 0 0 V20 0 0 0 0 0 0 0 0 0 V20 0 0 0 0 0 0 0 0 0 V20 0 0 0 0 0 0 0 0 0 V20 0 0 0 0 0 0 0 0 0 0 V20 0 0 0 0 0 0 0 0 0 V20 0 0 0 0 0 0 0 0 0 V20 V20 V20 V20 V20 V20 V20 V20 V20 V20</pre>	

Next, we create the formative matrix, which in this example is all zeros because all our observed variables are reflective (i.e. not formative). Recall, reflective means that the unobserved variables are theorized to *cause* the observed variables (scores); whereas formative means the unobserved variables are *caused by* the observed variables.

- 🕣												Tools Addins)							
Console	C:/Us	ers/jd	s0282	.UNT	/Des	ktop	/Wor	rkstuff,	/Jon_R	$ \approx$											_ (
form.																					
dimna			m. ma	atri	x)	<-	lis	t(c("f1",	"f2'	', "f	3", "	f4",	"f5	", "f	6"),	nam	es (p	ls.d	lf))	
form.															. 10						
	/2 V	3 V4	V5	V6	V/	V8	v9 '	V10	VII \	12 VI	13 VI	4 v15	V10	VI/	V18	V19	V20	V21 \	/22		
, 0	8		0	0	0	8	0	0	0	ŏ	0	0 0	0	0	0	0	0	0	0	0	
0	0		0	0	0	0	0	ő	ő	ŏ	-	0 0	-	-	ő	0	ő	ő	ő	ő	
ŏ	õ		ő	ŏ	ŏ	ŏ	õ	ŏ	ő	ŏ	•	0 0			ő	ŏ	ŏ	õ	ŏ	ŏ	
	ŏ	n n	ŏ	ŏ	ŏ	ŏ	ŏ	ŏ	ŏ	ŏ	-	ŏŏ			ŏ	ŏ	ŏ	ŏ	ŏ	ŏ	
5 0	õ	o o	ŏ	ŏ	ŏ	ŏ	ŏ	ŏ	ŏ	ŏ	-	õ õ	_	-	ŏ	ŏ	õ	ŏ	ŏ	ŏ	
50 60																					

Now we can combine all three matrices into a list object which specifies the structural model.

```
RStudio
File Edit Code View Plots Session Build Debug Profile Tools
                                                              Help
💽 🔹 🗲 🔚 🔒 🚔 🖌 🕞 🚱 🚱
                                             Addins -
                                                                                                                                 -0
  Console C:/Users/jds0282.UNT/Desktop/Workstuff/Jon R/
 > s.model <- list(inner = inner.matrix,
+ reflective = outer.matrix)
+ formative = form.matrix)
 +
+
> s.mo
$inner
   s.model
    f1
f2
f3
f4
f5
f6
 $formative
                      V6 V7
0 0
0 0
0 0
0 0
0 0
0 0
                            V1
0
0
0
0
0
0
        V2
0
0
0
0
0
0
           V3
0
0
0
0
0
               v4
                  v5
0
0
0
0
0
 f1
f2
f3
f4
f5
f6
               0 0 0 0 0 0 0
 >
```

The only other thing we need is the variance-covariance matrix of the observed variables.

RStudio

 File Edit Code View Plots Session Build Debug Profile Tools Help

 Go to file/function
 Addins

 Console C/Users/jds0282.UNT/Desktop/Workstuff/Jon_R/

 > cov.mat <- cov(pls.df)</td>

 >

 >

 >

Now we can apply the 'matrixpls' function using OLS Regression estimation as would be done with traditional PLS (e.g., the 'plspm' package).

RStudio

<pre>> mat.pls.1 # < just the parameter estimates and weights. matrixpls parameter estimates Est Far-f1 0.95780739 Far-f2 - 0.005780739 Far-f2 - 0.005780739 Far-f3 - 0.01649538 F5-f3 0.259157126 F5-f4 0.6191580 F5-f4 0.6194538 F5-f3 0.259157126 F5-f4 0.6194538 F5-f3 0.259157126 F5-f4 0.6194538 F5-f3 0.259157126 F5-f4 0.6194538 F2-w5 0.758034699 F2-w5 0.75803469 F2-w5 0.75803469 F2-w5 0.75803469 F2-w5 0.75803469 F2-w5 0.75803469 F2-w5 0.75803469 F2-w5 0.7580346 F2-w5 0.758074 F2-w5 0.758 F2-w5 0.75 F2-w5 0</pre>	Console C:/Users/jds0282.UNT/Desktop/Workstuff/Jon_R/ 🔗	-0
Est. 4.4-f1 0.5935519 74-f2 -0.005789733 74-f2 -0.005789733 74-f2 -0.005789733 74-f2 -0.005789733 75-f3 0.239157126 75-f4 0.239157126 75-f4 0.239157126 75-r4 0.50013234 1=-v4 0.70013234 1=-v4 0.702441380 72-w5 0.73034469 72-w5 0.73034469 72-w5 0.73034459 72-w5 0.73034459 72-w5 0.7303458 72-w5 0.7303458 72-w5 0.7303478 72-w5 0.7305878 72-w5	mat.pls.1 # < just the parameter estimates and weights.	
$ \begin{array}{cccccccccccccccccccccccccccccccccccc$		
$ \begin{aligned} f_{4}-f_{2} & -0.005789733 \\ f_{4}-f_{3} & -0.005789733 \\ f_{5}-f_{3} & 0.259157126 \\ f_{5}-f_{4} & 0.61961402 \\ f_{6}-f_{5} & 0.883304239 \\ f_{1}-x4 & 0.7724814345 \\ f_{1}-x2 & -0.506019294 \\ f_{1}-x4 & 0.772481390 \\ f_{2}-x4 & 0.86793723 \\ f_{2}-x4 & 0.956531212 \\ f_{3}-x41 & 0.779481738 \\ f_{3}-x41 & 0.779481738 \\ f_{3}-x41 & 0.779481738 \\ f_{3}-x41 & 0.77948173 \\ f_{3}-x41 & 0.779481738 \\ f_{3}-x41 & 0.779481738 \\ f_{3}-x41 & 0.77948173 \\ f_{3}-x41 & 0.879640677 \\ f_{4}-x415 & 0.871812387 \\ f_{5}-x419 & 0.881207693 \\ f_{5}-x42 & 0.634501106 \\ matring1s weights \\ f_{5}-x22 & 0.694501106 \\ matring1s weights \\ f_{5}-x22 & 0.694501106 \\ matring1s & 0.000000 & 0.0000000 & 0.261508 & 0.2580840 & 0.2945090 & 0.000000 & 0.000000 & 0.000000 \\ f_{3} & 0.000000 & 0.0000000 & 0.0000000 & 0.0000000 & 0.0000000 & 0.0000000 & 0.0000000 & 0.000000 & 0.000000 & 0.000000 & 0.000000 $		
F5-F3 0.259157126 F5-F4 0.619161402 F6-F5 0.883304239 F1		
$ \begin{array}{cccccccccccccccccccccccccccccccccccc$		
$ f_{1=-\sqrt{2}} - 0.506019294 \\ f_{1=-\sqrt{4}} - 0.51449558 \\ f_{1=-\sqrt{4}} - 0.574309274 \\ f_{2=-\sqrt{6}} 0.5874309274 \\ f_{2=-\sqrt{6}} 0.586793723 \\ f_{2=-\sqrt{8}} 0.537140330 \\ f_{3=-\sqrt{11}} 0.53693182 \\ f_{3=-\sqrt{11}} 0.53693182 \\ f_{3=-\sqrt{11}} 0.53693182 \\ f_{4=-\sqrt{14}} 0.573628892 \\ f_{4=-\sqrt{15}} 0.531693823 \\ f_{5=-\sqrt{19}} 0.5836698553 \\ f_{5=-\sqrt{12}} 0.5836698553 \\ f_{5=-\sqrt{21}} 0.5836698553 \\ f_{5=-\sqrt{21}} 0.58361166 \\ f_{6=-\sqrt{22}} 1.528041185 \\ f_{6=-\sqrt{22}} 1.528041185 \\ f_{6=-\sqrt{22}} 1.528041185 \\ f_{6=-\sqrt{23}} 0.5435100 \\ .0000000 0.0000000 0.0000000 0.0261058 0.2584864 0.2845544 0.0000000 0.0000000 0.0000000 0.0000000$	6~f5 0.883304239	
$ f_{1-x-y} = 0.54495588 \\ f_{2-x-y} = 0.78034499 \\ f_{2-x-y} = 0.79957877 \\ f_{2-x-y} = 0.79957877 \\ f_{2-x-y} = 0.79987877 \\ f_{2-x-y} = 0.79850788 \\ f_{2-x-y} = 0.794582707 \\ f_{2-x-y} = 0.78362892 \\ f_{2-x-y} = 0.886698533 \\ f_{2-x-y} = 0.98421185 \\ f_{2-x-y} = 0.18843140478 \\ f_{2-x-y} = 0.18843140478 \\ f_{2-x-y} = 0.18843140478 \\ f_{2-x-y} = 0.18843140478 \\ f_{2-x-y} = 0.528041185 \\ f_{2-x-y} = 0.528041185 \\ f_{2-x-y} = 0.18843140478 \\ f_{2-x-y} = 0.18842118 \\ f_{2-x-y} = 0.1884218 \\ f_{2-x-y} = 0.188421$		
$ f_{1=-\sqrt{4}} 0.7024419100 \\ f_{2=-\sqrt{6}} 0.784309274 \\ f_{2=-\sqrt{6}} 0.784309274 \\ f_{2=-\sqrt{6}} 0.86739723 \\ f_{2=-\sqrt{8}} 0.986739723 \\ f_{2=-\sqrt{8}} 0.986739723 \\ f_{2=-\sqrt{8}} 0.97560788 \\ f_{3=-\sqrt{11}} 0.079987877 \\ f_{3=-\sqrt{11}} 0.079987877 \\ f_{3=-\sqrt{11}} 0.653695152 \\ f_{3=-\sqrt{12}} 0.797560788 \\ f_{4=-\sqrt{13}} 0.875602892 \\ f_{4=-\sqrt{15}} 0.875602892 \\ f_{4=-\sqrt{15}} 0.8351692387 \\ f_{4=-\sqrt{15}} 0.8351692387 \\ f_{5=-\sqrt{19}} 0.835698533 \\ f_{5=-\sqrt{19}} 0.835698533 \\ f_{5=-\sqrt{19}} 0.833217693 \\ f_{6=-\sqrt{21}} 0.839698531 \\ f_{6=-\sqrt{21}} 0.839478311 \\ f_{6=-\sqrt{21}} 0.839478311 \\ f_{6=-\sqrt{21}} 0.843540166 \\ f_{6=-\sqrt{22}} 0.528041185 \\ f_{6=-\sqrt{22}} 0.528041185 \\ f_{6=-\sqrt{23}} 0.694501106 \\ f_{6} 0.000000 0.0000000 0.0000000 0.0000000 0.000000$		
$ \begin{array}{c} f_{2m} < V_{0} & 0, 783034469 \\ f_{2m} < V_{0} & 0.86793723 \\ f_{2m} < V_{0} & 0.86793723 \\ f_{2m} < V_{0} & 0.79987877 \\ f_{2m} < V_{11} & 0.63309152 \\ f_{2m} < V_{10} & 0.79987877 \\ f_{2m} < V_{11} & 0.797560788 \\ f_{4m} < V_{13} & 0.873622892 \\ f_{4m} < V_{13} & 0.873622892 \\ f_{4m} < V_{15} & 0.831192387 \\ f_{5m} < V_{10} & 0.83127693 \\ f_{5m} < V_{10} & 0.83127693 \\ f_{5m} < V_{10} & 0.836598533 \\ f_{5m} < V_{10} & 0.843310478 \\ f_{5m} < V_{20} & 0.843317693 \\ f_{5m} < V_{20} & 0.843317693 \\ f_{5m} < V_{20} & 0.644501106 \\ \\ matrixp1s weights \\ V & V & V & V & V & V & V & V & V \\ f_{1} & 0.328231 & -0.2425281 & 0.312154 & 0.356424 & 0.000000 & 0.000000 & 0.000000 & 0.000000 & 0.000000 \\ f_{2} & 0.000000 & 0.0000000 & 0.0000000 & 0.0000000 & 0.0000000 & 0.0000000 & 0.000000 \\ f_{2} & 0.000000 & 0.0000000 & 0.0000000 & 0.0000000 & 0.0000000 & 0.0000000 & 0.000000 & 0.000000 \\ f_{2} & 0.000000 & 0.0000000 & 0.0000000 & 0.0000000 & 0.0000000 & 0.0000000 & 0.0000000 & 0.000000 \\ f_{2} & 0.0000000 & 0.0000000 & 0.0000000 & 0.0000000 & 0.0000000 & 0.0000000 & 0.0000000 & 0.0000000 \\ f_{2} & 0.0000000 & 0.0000000 & 0.0000000 & 0.0000000 & 0.0000000 & 0.0000000 & 0.0000000 & 0.0000000 & 0.000000 & 0.0000000 & 0.000000 & 0.0000000 $	1=~v4 0.702945190	
f2=-v7 0.886793723 f2=-v7 0.95651122 f3=-v10 0.769987877 f3=-v11 0.653695152 f3=-v12 0.79750788 f4=-v13 0.87140380 f4=-v14 0.873682892 f4=-v14 0.873682892 f5=-v18 0.881930823 f5=-v18 0.883117693 f6=-v20 0.843140478 f6=-v21 0.894978351 f6=-v22 0.58401106 matrixpls weights v5 v6 v7 v8 v9 v0 f1 0.3928233 -0.2425821 0.3121514 0.365424 0.0000000 0.0000000 0.0000000 0.0		
$ \begin{array}{llllllllllllllllllllllllllllllllllll$		
$f_{3} = -\sqrt{10} 0.76987877$ $f_{3} = -\sqrt{10} 0.769887877$ $f_{3} = -\sqrt{11} 0.653895152$ $f_{3} = -\sqrt{12} 0.797560788$ $f_{4} = -\sqrt{13} 0.873682892$ $f_{4} = -\sqrt{13} 0.873682892$ $f_{4} = -\sqrt{13} 0.873682892$ $f_{5} = -\sqrt{10} 0.883217693$ $f_{5} = -\sqrt{10} 0.883217693$ $f_{5} = -\sqrt{10} 0.883217693$ $f_{5} = -\sqrt{10} 0.843340478$ $f_{6} = -\sqrt{22} - 0.528041185$ $f_{6} = -\sqrt{20} 0.000000 0.0000000 0.0000000 0.0000000 0.000000$	2=~v8 0.956551212	
f3=-v11 0.633691152 f3=-v12 0.797560788 f4=-v13 0.879640677 f4=-v13 0.831192387 f4=-v14 0.831192387 f5=-v12 0.831192387 f5=-v17 0.881930823 f5=-v18 0.883217693 f6=-v20 0.843340478 f6=-v21 0.894978331 f6=-v22 0.694501106 matrixpls wrights v1 v1 v2 v3 v4 v5 v6 v7 v8 v9 v10 f2 0.000000		
$ \begin{array}{l} f_{3} =12 & 0.797560788 \\ f_{4} =12 & 0.879640677 \\ f_{4} =14 & 0.873682892 \\ f_{4} =15 & 0.879640677 \\ f_{5} =17 & 0.881930823 \\ f_{5} =17 & 0.83698553 \\ f_{5} =18 & 0.83698553 \\ f_{5} =19 & 0.838217693 \\ f_{6} =22 & 0.843340478 \\ f_{6} =22 & 0.528041185 \\ f_{6} = -0.22 & 0.3121514 & 0.3656424 & 0.000000 & 0.000000 & 0.0000000 & 0.0000000 \\ 0.000000 & 0.000000 & 0.000000 & 0.000000 & 0.000000 & 0.000000 & 0.000000 & 0.000000 \\ f_{5} & 0.000000 & 0.000000 & 0.000000 & 0.000000 & 0.0000000 & 0.0000000 & 0.0000000 & 0.0000000 \\ f_{5} & 0.000000 & 0.0000000 & 0.0000000 & 0.0000000 & 0.0000000 & 0.0000000 & 0.0000000 & 0.0000000 \\ f_{5} & 0.000000 & 0.0000000 & 0.000000 & 0.0000000 & 0.0000000 & 0.0000000 & 0.0000000 & 0.0000000 \\ f_{5} & 0.000000 & 0.0000000 & 0.000000 & 0.0000000 & 0.0000000 & 0.0000000 & 0.0000000 & 0.0000000 \\ f_{5} & 0.0000000 & 0.0000000 & 0.0000000 & 0.0000000 & 0.0000000 & 0.0000000 & 0.0000000 & 0.0000000 \\ f_{5} & 0.0000000 & 0.0000000 & 0.000000 & 0.0000000 & 0.0000000 & 0.0000000 & 0.0000000 & 0.0000000 \\ f_{5} & 0.0000000 & 0.0000000 & 0.000000 & 0.0000000 & 0.0000000 & 0.0000000 & 0.0000000 & 0.0000000 \\ f_{5} & 0.0000000 & 0.0000000 & 0.000000 & 0.0000000 & 0.0000000 & 0.0000000 & 0.0000000 & 0.0000000 & 0.0000000 \\ f_{5} & 0.00000000$		
$ \begin{array}{c} f_4 = -\sqrt{14} & 0.873682892 \\ f_4 = -\sqrt{16} & 0.851192387 \\ f_4 = -\sqrt{16} & 0.794582707 \\ f_5 = -\sqrt{17} & 0.881930823 \\ f_5 = -\sqrt{19} & 0.835698553 \\ f_5 = -\sqrt{19} & 0.835698553 \\ f_5 = -\sqrt{21} & 0.894978351 \\ f_6 = -\sqrt{22} & 0.894978351 \\ f_6 = -\sqrt{22} & 0.694501106 \\ \\ \\ matrixpls weights & & & & & & & & & & & & & & & & & & &$	3=~v12 0.797560788	
$ \begin{array}{c} f_4 = -\sqrt{15} & 0.851192387 \\ f_4 = -\sqrt{16} & 0.794582707 \\ f_5 = -\sqrt{17} & 0.881930823 \\ f_5 = -\sqrt{12} & 0.83627693 \\ f_6 = -\sqrt{22} & 0.833217693 \\ f_6 = -\sqrt{22} & 0.528041185 \\ f_6 = -\sqrt{22} & 0.528041185 \\ f_6 = -\sqrt{23} & 0.694501106 \\ \end{array} $		
$ \begin{array}{c} f_{5} = - \sqrt{17} & 0.881930823 \\ f_{5} = - \sqrt{18} & 0.883698553 \\ f_{5} = - \sqrt{19} & 0.883217693 \\ f_{6} = - \sqrt{22} & 0.843340478 \\ f_{6} = - \sqrt{22} & 0.8439351 \\ f_{6} = - \sqrt{22} & 0.594501106 \\ \\ \hline matrixpls weights \\ \psi_1 & \chi_2 & \chi_3 & \chi_4 & \sqrt{5} & \sqrt{6} & \sqrt{7} & \sqrt{8} & \sqrt{9} & \sqrt{10} \\ f_1 & 0.3928233 & -0.2425821 & 0.3121514 & 0.3656424 & 0.000000 & 0.000000 & 0.000000 & 0.000000 & 0.000000 \\ f_2 & 0.000000 & 0.000000 & 0.000000 & 0.000000 & 0.2615038 & 0.2584804 & 0.2945904 & 0.2862354 & 0.000000 & 0.000000 \\ f_3 & 0.000000 & 0.000000 & 0.000000 & 0.000000 & 0.000000 & 0.000000 & 0.000000 & 0.000000 & 0.000000 \\ f_5 & 0.000000 & 0.000000 & 0.000000 & 0.000000 & 0.000000 & 0.000000 & 0.000000 & 0.000000 & 0.000000 \\ f_5 & 0.000000 & 0.000000 & 0.000000 & 0.000000 & 0.000000 & 0.000000 & 0.000000 & 0.000000 & 0.000000 \\ f_5 & 0.000000 & 0.000000 & 0.000000 & 0.000000 & 0.000000 & 0.000000 & 0.000000 & 0.000000 & 0.000000 \\ f_1 & 0.000000 & 0.000000 & 0.000000 & 0.000000 & 0.000000 & 0.000000 & 0.000000 & 0.000000 & 0.000000 \\ f_1 & 0.000000 & 0.000000 & 0.000000 & 0.000000 & 0.000000 & 0.000000 & 0.000000 & 0.000000 & 0.000000 & 0.000000 \\ f_2 & 0.000000 & 0.000000 & 0.000000 & 0.000000 & 0.000000 & 0.000000 & 0.000000 & 0.000000 & 0.000000 & 0.000000 \\ f_2 & 0.000000 & 0.000000 & 0.000000 & 0.000000 & 0.000000 & 0.000000 & 0.000000 & 0.000000 & 0.000000 & 0.000000 \\ f_2 & 0.000000 & 0.000000 & 0.000000 & 0.000000 & 0.000000 & 0.000000 & 0.000000 & 0.000000 & 0.000000 \\ f_3 & 0.297447 & 0.282879 & 0.0000000 & 0.0000000 & 0.000000 & 0.0000000 & 0.0000000 & 0.000000 & $		
<pre>F5=-v18 0.836608553 F5=-v19 0.88360478 F5=-v21 0.88340478 F6=-v22 0.843340478 F6=-v22 0.528041185 F6=-v23 0.694501106 matrixpls weights v1 v2 v3 v4 v5 v6 v7 v8 v9 v10 F1 0.3928233 -0.2425821 0.3121514 0.3656424 0.000000 0.000000 0.000000 0.0000000 0.000000</pre>		
$ \begin{array}{l c c c c c c c c c c c c c c c c c c c$		
<pre>F6=-v22 -0.528041185 f6=-v23 0.694501106 matrixpls weights</pre>		
f6 = -v23 0.694501106		
$ \begin{array}{cccccccccccccccccccccccccccccccccccc$		
$ \begin{array}{cccccccccccccccccccccccccccccccccccc$	matrixnls weights	
f2 0.0000000 0.0000000 0.0000000 0.0000000		
$ \begin{array}{cccccccccccccccccccccccccccccccccccc$		
f4 0.000000 0.000000 0.000000 0.000000 0.000000		
$ \begin{array}{cccccccccccccccccccccccccccccccccccc$		
v11 v12 v13 v14 v15 v16 v17 v18 v19 v20 v21 f1 0.0000000 <t< td=""><td></td><td></td></t<>		
f1 0.000000 0.000000 0.000000 0.000000 0.000000		v21
f3 0.2974447 0.2829879 0.0000000 0.000000 0.0000000 0.000000 0.000000		
F4 0.000000 0.000000 0.2630766 0.276333 0.2737062 0.3150948 0.000000 0.0000000 0.0000000 0.0000000 F5 0.000000 0.000000 0.0000000 0.0000000 0.000000		
f6 0.000000 0.000000 0.000000 0.000000 0.000000		
v22 v23 f1 0.000000 0.000000 f2 0.000000 0.000000 f3 0.000000 0.000000 f4 0.000000 0.000000 f5 0.000000 0.000000		
f1 0.000000 0.000000 f2 0.000000 0.000000 f3 0.000000 0.000000 f4 0.000000 0.000000 f5 0.000000 0.000000		.649
f3 0.000000 0.000000 f4 0.000000 0.000000 f5 0.000000 0.000000		
f4 0.000000 0.000000 f5 0.000000 0.000000		
f5 0.0000000 0.0000000		
F6 -0.2416281 0.2964546		
	6 -0.2416281 0.2964546	

A much more thorough presentation of output can be seen by using either the 'summary' function or the 'attributes' function; however, the output of each is only partially presented here due to their size. The image below shows what is returned when applying the 'names' function to the 'summary', as well as the 'attributes', of the 'matrixpls' object. Using the various returned names one can then extract relevant elements of the 'matrixpls' object (e.g., summary(mat.pls.1)\$gof can be used to extract the absolute goodness of fit).

```
RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
💽 🔹 🥣 🖌 🔒 👘 🥌 🖉 🚱 🖓 🚱
                                               Addins 🗸
                                                                                                                                       -0
  Console C:/Users/ids0282.UNT/Desktop/Workstuff/Jon R/
       0.3825689
                                 0.33/54/8 -0.1810999
0.3824874 -0.2050156
                    0.2968869
                                                            0.3319341
       0.5151474
                    0.3361843
 v4
                                                            0.3768826
 v5
       0.6567599
                    0.3635502
                                 0.4474943 -0.3765618
                                                            0.4162175
       0.5574789
                                 0.3955072 -0.3341406
 v6
                    0.3206306
                                                            0.3686761
                                 0.4250091 -0.3649024
 ν7
       0.6454113
                    0.3415324
                                                            0.3997538
 v8
       0.7182835
0.5161412
                    0.3868567
                                 0.4780302 -0.4051640
0.4321325 -0.2730097
                                                            0.4464003
                    0.3768651
                                                            0.4227097
 v9
 v10
       0.4027146
                    0.2997746
                                 0.3465838 -0.2204397
                                                            0.3461785
       0.2943062
                                 0.2893979 -0.1842903
                    0.2499377
 v11
                                                            0.2901381
                    0.3728155
 v12
       0.4483266
                                 0.4251159 -0.2673453
                                                            0.4098848
 v13
       0.6933074
                    0.4721942
                                 0.5313198 -0.2268184
                                                            0.5521096
 v14
       0.6968462
                    0.4901985
                                 0.5502218 -0.2385576
                                                            0.5658307
 v15
       0.5880577
                    0.4380973
                                 0.4941580 -0.2076955
                                                            0.5187508
       0.4554743
 v16
                    0.3025109
                                 0.3483253 -0.1272308
                                                            0.3965875
       0.7917769
0.7072813
 v17
                    0 5688732
                                 0 5990583 -0 4149797
                                                            0 5073431
 v18
                                 0.6878542 -0.4642138
                    0.6523212
                                                            0.5740701
                                 0.7284872 -0.4914331
0.7357312 -0.5374155
 v19
       0.7800735
                    0.6908407
                                                            0.6078416
 v20
       0.6250233
                    0.7112232
                                                            0.6407889
 v21
       0.6679474 0.7738117
                                 0.8009862 -0.5822432
                                                           0.6957044
 v22 -0.3324695 -0.3532215 -0.3629277 0.2788275
v23 0.4757884 0.5306129 0.5474225 -0.4080759
                                                           -0.3253745
                                                           0.4823318
  Residual-based fit indices
                                          value
 Communality
                                     0.6368069
 Redundancy
                                     0.2512407
 SMC 0.7673364
RMS outer residual covariance 0.4620969
 RM5 inner residual covariance 0.4625171
 SRMR
                                     0.4424241
 SRMR (Henseler)
                                     0.4256259
  Absolute goodness of fit: 0.6990315
  Composite Reliability indices
                      f7
                                              f4
 0.7482106 0.9300959 0.8578344 0.9125638 0.9012123 0.8354544
  Average Variance Extracted indices
f1 f2 f3
                                             f4
                                                          f5
 0.4320324 0.7697383 0.6037969 0.7232449 0.7526466 0.5683422
  AVE - largest squared correlation
 f1 f2 f3 f4 f5 f6
-0.46326737 0.23708968 0.24596341 -0.17205485 -0.02757973 -0.21188421
  Heterotrait-monotrait matrix
                                     f3
                         f2
                                                 f4
                                                             f5
                                                                         f6
             f1
 f1 0.0000000
 f2 0.7448018 0.0000000
 f3 1.0148647 0.5285766 0.0000000
f4 1.6118478 0.5690019 0.5435413 0.0000000
 f5 1.1326688 0.7291690 0.5982384 0.7561813 0.0000000
f6 1.5362715 0.6651057 0.7969583 1.0373163 1.1797596 0.0000000
      ames(summary(mat.pls.1))
"estimates" "effects" "r2"
                                                 "residuals" "gof"
 [1]
                                                                              "cr"
                                                                                            "ave"
                                                                                                          "htmt'
  >______ errects" "r2"
names(attributes(mat.pls.1))
[1] "names" "S" "E"
[9] "inner" "reflective" "for
                                                                                    "history"
                                                      "iterations" "converged"
                                                                                                                    "IC"
"class"
                      "reflective" "formative" "Q
                                                                                                     'call"
                                                                      'w'
                                                                                      'model
```

Next, we re-specify the same model using the 'lavaan' package style of model specification syntax. This style of model specification syntax is much more intuitive and requires fewer lines (of code); as well as fewer objects in the workspace (i.e. not 3 matrices).

```
Image: StudioFileEditCodeViewPlotsSessionBuildDebugProfileToolsHelpImage: ToolsImage: ToolsImage: ToolsImage: ToolsHelpImage: ToolsHelpImage: ToolsImage: ToolsImage: ToolsImage: ToolsHelpImage: ToolsImage: ToolsHelpImage: ToolsImage: ToolsImage
```

RStudio

Next, we will fit the same model, as re-specified using 'lavaan' syntax, but we will use GSCAc estimation and 2-stage least squares. Note, only partial output is displayed from the 'summary' function.

-0

File Edit Code View Plots Session Build Debug Profile Tools Help 🐑 🔹 🥶 📲 🔝 🚔 🚺 🏕 Go to file/function 🛛 😁 🕶 Addins 🗸 Console C:/Users/jds0282.UNT/Desktop/Workstuff/Jon_R/ 🔅 -0 > mat.pls.2 <- matrixpls(5 = cov.mat, + model = lavaan.s.model, model = lavaan.s.model, disattenuate = TRUE, innerEstim = innerEstim.gsca, outerEstim = outerEstim.gsca, parametersInner = estimator.tsls, parametersReflective = estimator.cfaLoadings, validateInput = TRUE, standardize = TRUE) ++ + + + + > summary(mat.pls.2) matrixpls parameter estimates Est. 0.954709360 F4~F1 F4~F2 F4~F3 0.00221247 -0.012776329 F5~F3 0.219381631 F5~F4 0.680308179 E6~E5 0.821313736 F1=~V1 F1=~V2 0.774814345 -0.506019294 F1=~V3 F1=~V4 0.614495588 0.702945190 0.874309274 0.783034469 F2=~v5 F2=~v6 F2=~V7 0.886793723 F2=~V8 0.956551212 F3=~v9 0.871140380 F3=~v10 F3=~v11 0.769987877 0.653695152 0.797560788 F3=~v12 F4=~v13 F4=~v14 0.873682892 F4=~v15 0.85119238 F4=~v16 0.794582707 F5=~v17 0.881930823 F5=~v18 0.836698553 F5=~V19 F6=~V20 0.883217693 F6=~v21 0.894978351 F6=~v22 -0.52804118 F6=~v23 0.694501106 matrixpls weights F3 0.000000 F4 0.000000 F5 0.000000 F6 0.000000 v11 v12 v13 v14 v15 v16 v17 v18 v19 v20 V21 V22 V23 F1 0.0000000 0.000000 0.0000000

So, now we can do a comparison of the coefficients based on the two estimation techniques used on the same data and the same model — keep in mind with this simulated data there are very small differences in the statistics produced.

] 🗸 🥣 🔹 🔚 💭 🖶 🚺 🌈 Go to file/function 🛛 🛛 🔀 🔸 Addins 👻	
Console C:/Users/jds0282.UNT/Desktop/Workstuff/Jon_R/	
<pre>cbind(PLS = mat.pls.1, GSCAc = mat.pls.2)</pre>	
PLS GSCAC	
4~f1 0.959585619 0.954709360	
4~f2 -0.005789733 0.002212477	
4~f3 -0.016649558 -0.012776329	
5~f3 0.259157126 0.219381631 5~f4 0.619161402 0.680308179	
6~f5 0.883304239 0.821313736	
1=~v1 0.774814345 0.774814345	
1=~v2 -0.506019294 -0.506019294	
1=~V3 0.614495588 0.614495588	
1=~v4 0.702945190 0.702945190	
2=~v5 0.874309274 0.874309274	
2=~v6 0.783034469 0.783034469	
2=~v7 0.886793723 0.886793723	
2=~v8 0.956551212 0.956551212	
3=~v9 0.871140380 0.871140380	
3=~v10 0.769987877 0.769987877 3=~v11 0.653695152 0.653695152	
3=~v11 0.653695152 0.653695152 3=~v12 0.797560788 0.797560788	
4=~v13 0.879640677 0.879640677	
4=~v14 0.873682892 0.873682892	
4=~v15 0.851192387 0.851192387	
4=~v16 0.794582707 0.794582707	
5=~v17 0.881930823 0.881930823	
5=~v18 0.836698553 0.836698553	
5=~v19 0.883217693 0.883217693	
6=~v20 0.843340478 0.843340478	
6=~v21 0.894978351 0.894978351	
6=~v22 -0.528041185 -0.528041185	
6=~v23 0.694501106 0.694501106	
1=+v1 0.392823317 0.390177044 1=+v2 -0.242582070 -0.249234179	
1=+v3 0.312151407 0.323002783	
1=+v4 0.365642375 0.352330123	
2=+v5 0.261503787 0.278376467	
2=+v6 0.258480437 0.259794071	
2=+v7 0.294590385 0.277288633	
2=+v8 0.286235427 0.285619902	
3=+v9 0.302373538 0.310613521	
3=+v10 0.315367959 0.302975092	
3=+v11 0.297444708 0.276602284	
3=+v12 0.282987855 0.305783594	
4=+v13 0.263076566 0.265169073	
4=+v14 0.276333013 0.268466927	
4=+v15 0.273706230 0.243131683 4=+v16 0.315094753 0.352841361	
4=+v16 0.315094753 0.352841361 5=+v17 0.369771041 0.421717572	
5=+v18 0.351571788 0.308560886	
5=+v19 0.371769210 0.362454977	
6=+v20 0.333790371 0.331341786	
6=+v21 0.351648995 0.359036091	
6=+v22 -0.241628111 -0.240094050	
6=+v23 0.296454644 0.291934415	

We can also compare the measurement model's composite reliability estimates (i.e. the "Q" statistic) from each method of estimation. Note, the "Q" statistic produced is analogous to coefficient alpha from traditional item evaluation.

If interested in obtaining the Q2 predictive relevance statistic(s) you first need to run the cross–validation function which mimics the blindfolding procedure used with the 'semPLS' function of the package by the same name. Then, apply the 'q2' function to the object returned by the cross-validation.

```
RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
💽 🔹 🥣 💼 🔒 🚔 🚺 🏕 Go to file/function
                                       Addins -
 Console C:/Users/jds0282.UNT/Desktop/Workstuff/Jon_R/
                                                                                                             -0
 > cv.blindfold <- matrixpls.crossvalidate(pls.dd</p>
                         model = lavaan.s.model,
blindfold = TRUE,
predictionType = "redundancy",
 + groups = 4)
> q2(pls.df, cv.blindfold, lavaan.s.model)
 Q2 predictive relevance statistics
  Overall Q2
 0.4390654
 Block Q2
                 F2
 F1 F2 F3 F4 F5 F6
0.2538969 0.5518664 0.4846634 0.5097722 0.4361636 0.3985074
 Indicator 02
 v23
 0.3666231
```

There are many benefits to using the 'matrixpls' package (Ronkko, 2016a) rather than the 'plspm' package (Sanchez, Trinchera, & Russolillo, 2016) or the 'semPLS' package (Monecke, & Leisch, 2012) for fitting structural models when SEM cannot be used. Obviously, the greatest benefits of the 'matrixpls' package is the ability to use multiple new and more robust estimation methods; only one such combination of estimation techniques was used above. The 'matrixpls' function is also more computationally efficient (Ronkko, 2016c, p. 2) and offers more flexibility with respect to the types of models that can be fit. Consider the limitations on the specification of model matrices with traditional PLS packages. Those packages require the following matrix restrictions: "the inner must be a lower triangular matrix, reflective must have exactly one non-zero value on each row and must have at least one non-zero value on each column, and formative must only contain zeros" (Ronkko, 2016c, p. 4). The 'matrixpls' function has two restrictions; all matrices must be binary and the inner must have zeros on the diagonal (Ronkko, 2016c). Another benefit of the 'matrixpls' function is the ability to specify a model using the 'lavaan' package (Yves, 2012a) model specification syntax which is highly intuitive; interested readers should review the 'lavaan' package documentation available at CRAN (see Yves, 2012b). Lastly, another major benefit to using the 'matrixpls' package is a function for doing Monte Carlo simulations of a 'matrixpls' object.

A version of the R script used in this article can be found on the R&SS Do-It-Yourself Introduction to

 R^2 website at the bottom of the Module 9 section.

Until next time; everybody look what's going down

References and Resources

Bentler, P. M., & Huang, W. (2014). On components, latent variables, PLS and simple methods: Reactions to Rigdon's rethinking of PLS. *Long Range Planning*, 47(3), 138 - 145.

Dijkstra, T. (1983). Some comments on maximum likelihood and partial least squares methods. *Journal of Econometrics*, 22, 67 - 90.

Dijkstra, T. K., & Henseler, J. (2015a). Consistent partial least squares path modeling. *MIS Quarterly*, 39(2), 297 - 316.

Dijkstra, T. K., & Henseler, J. (2015b). Consistent and asymptotically normal PLS estimators for linear structural equations. *Computational Statistics and Data Analysis*, *81*, 10 - 23.

Faulk, R. R., & Miller, N. B. (1992). A primer for soft modeling. Akron, OH: The University of Akron.

Fornell, C., & Bookstein, F. L. (1982). Two structural equation models: LISREL and PLS applied to consumer exit-voice theory. *Journal of Marketing Research*, *19*, 440-452.

Huang, W. (2013). "PLSe: Efficient Estimators and Tests for Partial Least Squares." Doctoral dissertation, Los Angeles: University of California.

Hwang, H., & Takane, Y. (2004). Generalized structured component analysis. *Psychometrika*, 69(1), 81 -99.

Hwang, H., & Takane, Y. (2014). *Generalized structured component analysis: A component-based approach to structural equation modeling*. New York: CRC Press / Taylor & Francis Group.

Monecke, A., & Leisch, F. (2012). semPLS: Structural Equation Modeling Using Partial Least Squares. *Journal of Statistical Software*, 48(3), 1-32. URL http://www.jstatsoft.org/v48/i03/. Documentation available at CRAN:

https://cran.r-project.org/web/packages/semPLS/index.html

Rigdon, E. E. (2012). Rethinking partial least squares path modeling: In praise of simple methods. *Long Range Planning*, *45*, 341 - 358.

Ronkko, M. (2016a). matrixpls: Matrix-based Partial Least Squares Estimation. R package version 1.0.4. Documentation available at CRAN:

https://cran.r-project.org/web/packages/matrixpls/index.html

Ronkko, M. (2016b). Package matrixpls manual. Available at CRAN: https://cran.r-project.org/web/packages/matrixpls/matrixpls.pdf

²http://bayes.acs.unt.edu:8083/BayesContent/class/Jon/R_SC/

Ronkko, M. (2016c). Package matrixpls vignette: Introduction to matrixpls. Available at CRAN: https://cran.r-project.org/web/packages/matrixpls/vignettes/matrixpls-intro

Sanchez, G. (2013). *PLS Path Modeling with R* (Trowchez Editions). Berkeley, 2013. Available at: http://gastonsanchez.com/PLS_Path_Modeling_with_R.pdf

Sanchez, G., Trinchera, L., & Russolillo, G. (2016). Package 'plspm' with documentation, including package manual, available at CRAN: https://cran.r-project.org/web/packages/plspm/index.html

Starkweather, J. (2011). An alternative modeling strategy: Partial least squares. As published in the *RSS Matters* (now called *Research Matters*) column of University Information Technology's *Benchmarks* online magazine. Available at:

http://bayes.acs.unt.edu:8083/BayesContent/class/Jon/Benchmarks/PLS_JDS_Jul

Vinzi, V. E., Chin, W. W., Hensler, J., & Wang, H. (2010). *Handbook of partial least squares: Concepts, methods, and applications.* New York: Springer.

Wold, H. O. A. (1965). A fixed-point theorem with econometric background, I-II. *Arkiv for Matematik,* 6, 209 - 240.

Wold, H. O. A. (1966a). Estimation of principal components and related methods by iterative least squares. In P. R. Krishnaiah (Ed.), *Multivariate Analysis* (pp. 391 - 420). New York: Academic Press.

Wold, H. O. A. (1966b). Nonlinear estimation by iterative least squares procedures, in: E. N. David (Ed.), *Research papers in statistics: Festschrift for J. Neyman* (pp. 411 - 444). New York: Wiley.

Wold, H. O. A. (1973). Nonlinear iterative partial least squares (NIPALS) modeling: Some current developments. In P. R. Krishnaiah (Ed.), *Multivariate Analysis* (pp. 383 - 487). New York: Academic Press.

Wold, H. O. A. (1982). Soft modeling: The basic design and some extension. In K. G. Joreskog and H. Wold (Eds.), *Systems under Indirect Observations II* (pp. 1 - 54). Amsterdam: North-Holland.

Yves, R. (2012a). lavaan: An R Package for Structural Equation Modeling. *Journal of Statistical Software*, 48(2), 1-36. URL http://www.jstatsoft.org/v48/i02/.

Yves, R. (2012b). Package 'lavaan' manual which is available at CRAN: https://cran.r-project.org/

This article was last updated on January 23, 2017.

This document was created using LATEX