Reliability practical

Based on Najera et al (Forthcoming): https://hectornajera83.github.io/book/preface.html

Dr. Hector Najera

18 July 2019

So first we need to set our working directory (where our files are located):

```
setwd("C:/Proyectos Investigacion/Lectures/Bristol Poverty Workshop 2019/Data")
#install.packages("psych")
#install.packages("lavaan")
#install.packages("ltm")
#install.packages("plyr")
#install.packages("ggplot")


#library(psych)
#library(lavaan)
#library(ltm)
#library(plyr)
```

To introduce the idea of reliability we will use the data set “Rel_MD_data_1_1.dat”. This is simulated data of a higher-order multidimensional measure of poverty (n=5000). The measure has nine indicators in total, distributed evenly in three dimensions.
First we are going to inspect our data:

```
Rel_MD_1<-read.table("Rel_MD_data_1_1.dat")
colnames(Rel_MD_1)<-c("x1","x2","x3","x4","x5","x6",
                      "x7","x8","x9","x10","x11",
                      "resources","educ_yr","occupation","hh_members","class")
str(Rel_MD_1)
## 'data.frame':    5000 obs. of  16 variables:
##  $ x1        : int  1 0 0 1 1 1 0 0 1 0 ...
##  $ x2        : int  1 0 0 1 0 0 0 0 0 0 ...
##  $ x3        : int  1 0 0 0 0 0 0 0 0 0 ...
##  $ x4        : int  1 0 1 0 0 0 1 1 1 0 ...
##  $ x5        : int  0 0 0 0 0 0 0 0 1 0 ...
##  $ x6        : int  0 0 0 0 0 0 1 0 1 0 ...
##  $ x7        : int  0 0 0 1 0 0 0 0 1 0 ...
##  $ x8        : int  0 0 0 0 0 0 0 0 1 0 ...
##  $ x9        : int  0 0 0 0 0 0 0 0 1 0 ...
##  $ x10       : int  0 0 0 0 1 0 0 1 0 1 ...
##  $ x11       : int  0 0 0 0 1 0 0 1 0 0 ...
##  $ resources : num  3277 7509 7184 1574 2210 ...
##  $ educ_yr   : int  6 15 8 6 9 13 12 7 7 6 ...
##  $ occupation: int  4 2 5 2 5 5 6 7 5 3 ...
##  $ hh_members: int  5 1 2 7 4 4 2 4 1 3 ...
##  $ class     : int  2 1 1 2 2 1 1 2 2 2 ...
```

Below we ask R to shown the first ten rows of our data set (the nine deprivation vars). Prior the analysis we need to conduct some basic inspections. We can compute a simple deprivation score/count called `ds` by applying `rowSums()` to the first nine items (x1-x9).

```
Rel_MD_1[1:10,1:11]
##    x1 x2 x3 x4 x5 x6 x7 x8 x9 x10 x11
## 1   1  1  1  1  0  0  0  0  0   0   0
## 2   0  0  0  0  0  0  0  0  0   0   0
## 3   0  0  0  1  0  0  0  0  0   0   0
## 4   1  1  0  0  0  0  1  0  0   0   0
## 5   1  0  0  0  0  0  0  0  0   1   1
## 6   1  0  0  0  0  0  0  0  0   0   0
## 7   0  0  0  1  0  1  0  0  0   0   0
## 8   0  0  0  1  0  0  0  0  0   1   1
## 9   1  0  0  1  1  1  1  1  1   0   0
## 10  0  0  0  0  0  0  0  0  0   1   0
```

Now we can check the proportion of people deprived of each indicator as follows. First, we will ask R to estimate the `mean()` for all indicators and store this information on `dep_prop`. Then we will request R to round them up but only for the nine deprivation indicators. We see that 50% of people in this sample lack x1, for example.

```
dep_prop<-unlist(lapply(Rel_MD_1, function(x) mean(x)))
dep_prop<-round(dep_prop[1:9]*100,0)
dep_prop
## x1 x2 x3 x4 x5 x6 x7 x8 x9
## 50 29 16 49 29 16 45 26 16
```

Exploratory (non-model based) estimation of overall reliability

Now that we have familiarised with the data ourselves we can proceed to check the reliability of this scale. In this section we will focus on the case where we do not have a poverty measure model (i.e. we have a list of indicators but we do not have a structure). Reliability concerns with the homogeneity of a scale and its capacity to produce consistent rankings of a population. We

will start by estimating the overall reliability of our scale using the psych package (Revelle 2014). This is a comprehensive R-package to estimate different reliability statistics (α, β, ω and ωh) under different changing conditions. The psych package can be used for exploratory and confirmatory settings for both unidimensional and multidimensional measures. This book focuses on confirmatory measurement models and to introduce the estimation of overall reliability we will rely on the simplest way to estimate the homogeneity of a scale using the simulated data set. This will be further developed, and the next section shows how psych interacts with another R-package lavaan to estimate ω and ωh from a confirmatory factor model (Rosseel 2012).
The pysch package permits estimating α and ω using the same function (omega). The package has several options, but we know that there are three dimensions and one higher order factor and these values match the defaults of the omega function. It is important to bear in mind that in this simple case the value of ω is approximated with an Exploratory Factor Analysis (EFA). Below is shown how to do it with a confirmatory model.
After applying the omega() to our nine indicators, there will be different objects that store information with the results of the analysis. We will focus on the overall estimate of α and ω as here we are interested in knowing the homogeneity of our scale. To extract the relevant estimates we will produce a data.frame that contains the results from omega_exp1.

```
# install.packages("psych")
require(psych)
omega_exp1<-omega(Rel_MD_1[,c(3:9)])
rel_uni_exp<-data.frame(omega_exp1=omega_exp1$omega.tot,
                        alpha=omega_exp1$alpha)
rel_uni_exp
##   omega_exp1     alpha
## 1  0.8599319 0.8127129
```

e can appreciate below that both values are high ($\geq .8$) (See section @ref(Chapter-3-measuresrel) for an explanation) and suggest that the scale is highly reliable. In this case, $\alpha < \omega$ indicating that this scale violates τ equivalence (equality of loadings).
Both α and ω are easily estimated with the psych package. However, the previous example was straightforward in that all the indicators are well-behaved. Thus, to gain a deeper understanding of reliability and population classification we will check what happens when one has indicators that reduce reliability. This can be done by adding noise to our measure. We will generate two uncorrelated indicators and substitute x10 and x11 for the indicators x1 and x2.
Then we can apply omega() to the new matrix that includes x10 and x11 and excludes x1 and x2. Reliability has drop slightly but enough to raise concerns as both ω and α are below the rules of thumb drawn from a Monte Carlo experiment.

```
omega_unr_exp<-omega(Rel_MD_1[,c(3:11)])
unrel_uni_exp<-data.frame(omega_exp=omega_unr_exp$omega.tot,
                          alpha=omega_unr_exp$alpha)
unrel_uni_exp
##   omega_exp    alpha
## 1  0.799405 0.738685
```

What is the impact of introducing the two uncorrelated indicators? From theory is known that losses in reliability affect the consistency of population classification. We can check if this theory holds by looking at the correlation of different rankings that are produce from different measures.

For this experiment, first, we will estimate the omega values using different combinations of items (in all case we have the seven items from the reliable measure x1-x9).

```
omega_exp2<-omega(Rel_MD_1[,c(3:9)])

omega_exp3<-omega(Rel_MD_1[,c(1,2,4,5,7,8)])

omega_exp4<-omega(Rel_MD_1[,c(2,3,5,6,8,9)])

omega_exp5<-omega(Rel_MD_1[,c(1,3,4,6,7,9)])



omegas_exp<-data.frame(omega_exp1=omega_exp1$omega.tot,

                       omega_exp2=omega_exp2$omega.tot,

                       omega_exp3=omega_exp3$omega.tot,

                       omega_exp4=omega_exp4$omega.tot,

                       omega_exp5=omega_exp5$omega.tot,

                       omega_unrel=omega_unr_exp$omega.tot)
```

We then can compare the omega values of each measure. The theory holds for this example. We see that the lowest reliability scale is the one that incorporates V1 and V2. The measures with only seven items have higher reliability. This is a very important lesson as poverty researchers sometimes keep unreliable indicators in their scales and the consequence will be a heavy loss in reliability.

```
t(omegas_exp)

##                  [,1]

## omega_exp1  0.8599319

## omega_exp2  0.8599319

## omega_exp3  0.8779016

## omega_exp4  0.8543497

## omega_exp5  0.8441886

## omega_unrel 0.7994050
```

The second prediction of reliability theory is that the population orderings are consistent for high reliability values. One way to check this is by estimating the correlation among the different deprivation scores. Again, the theory holds for this simple exercise, the measure with higher ω are highly correlated. The correlation of the unreliable measure seems still high, however, when $\omega < .8$ we could expect to see a classification error $> 5\%$ which might be very worrying when put into perspective. If the poverty rate is 20% and the classification error is 5% it would mean that potentially a 25% of the poor are mistakenly classified (Nájera 2018).
Reliability will be affected by the items that we keep in our scale. To gain deeper understanding of how different sets of items affect reliability we will use different combinations of items and then we will compute its reliability. First we will star by producing a deprivations score by using taking the sum of the nine reliable items with `rowSums`.

```
Rel_MD_1$ds<-rowSums(Rel_MD_1[,c(1:9)])
```

We can inspect the distribution of this deprivation score by plotting it (Figure @ref(fig:depscore)) as follows:

```r
require(ggplot2)
ggplot(Rel_MD_1, aes(ds)) +
    geom_histogram() + theme_bw() + labs(x = "Deprivation score") +
  scale_x_continuous(breaks = seq(0, 9, by = 1))
```

This is the histogram of the deprivation score. It shows the number of people by the equally weighted deprivation count.

We see that our deprivation score looks inflated in the right and deflated on the left. This is the expected distribution of a deprivation score in that the majority of the population tends to lack fewer items. This could look very different in very poor countries where most of the population like the majority of basic needs.

To see the relationship between reliability and different scores, we will compute five more scores using different combinations of items. We will see if reliability is affected by the number of items. The correlation matrix shows that if we keep reliable items only, we will have very high correlation with the perfect measure, which in this case is the measure with the nine items (ds). However, we see that, the measure with unreliable items leads to a drop of .93. This is not too bad as ω remains high for this scale, but if we drop a couple of items we would see a major drop (.87) (see ds_unrel2).

```r
Rel_MD_1$ds_r2<-rowSums(Rel_MD_1[,c(3:9)])
Rel_MD_1$ds_r3<-rowSums(Rel_MD_1[,c(1,2,4,5,7,8)])
Rel_MD_1$ds_r4<-rowSums(Rel_MD_1[,c(2,3,5,6,8,9)])
Rel_MD_1$ds_r5<-rowSums(Rel_MD_1[,c(1,3,4,6,7,9)])
Rel_MD_1$ds_ur<-rowSums(Rel_MD_1[,c(3:9,10,11)])
Rel_MD_1$ds_ur2<-rowSums(Rel_MD_1[,c(3:7,10,11)])


ds.m<-(Rel_MD_1[,c(17:23)])
ds.cor<-cor(ds.m)
ds.cor
##               ds     ds_r2     ds_r3     ds_r4     ds_r5     ds_ur
## ds     1.0000000 0.9684719 0.9764524 0.9523882 0.9733462 0.9272651
## ds_r2  0.9684719 1.0000000 0.9284786 0.9370200 0.9485305 0.9520979
## ds_r3  0.9764524 0.9284786 1.0000000 0.8875033 0.9371505 0.8899044
## ds_r4  0.9523882 0.9370200 0.8875033 1.0000000 0.8876694 0.8941369
## ds_r5  0.9733462 0.9485305 0.9371505 0.8876694 1.0000000 0.9099612
## ds_ur  0.9272651 0.9520979 0.8899044 0.8941369 0.9099612 1.0000000
## ds_ur2 0.8700863 0.8860309 0.8437632 0.8124119 0.8690256 0.9628708
##           ds_ur2
## ds     0.8700863
## ds_r2  0.8860309
## ds_r3  0.8437632
## ds_r4  0.8124119
## ds_r5  0.8690256
## ds_ur  0.9628708
## ds_ur2 1.0000000
```

Computation using R

We will continue using our simulated data (“Rel_MD_data_1_1.dat”). As a remainder we have stored the data on the object `Rel_MD_1` (see above). So the first question we need to answer is *What is the structure of our measure?*. That is, how many dimensions and how many indicators (and which indicators) for each dimension. In this case, we will assume that our theory tells us that we have a higher-order factor (poverty), three subfactors (dimensions of poverty) and three indicators for each dimension. With that in mind, we need to tell `lavaan` how our model looks like and, because ω and ωh is easier to compute from Schmid and Leiman (1957) transformation, we will specify such type of model directly rather than a nested model. To do so, we will store the model on the object `MD_model`. We are saying something very simple. First, we are saying that the higher-order factor *h* has nine manifest variables (x1-x9). Then we say that each dimension has three outcome variables. For example, F1 has x7, x8 and x9. Then we tell `lavaan` that our factor variance are equal to zero for identification and being consistent with the Schmid and Leiman (1957) transformation.

```
#Omega from Sem
library(lavaan)
# We first specify the model
MD_model <- ' h =~ +x1+x2+x3+x4+x5+x6+x7+x8+x9
                F1=~  + x7 + x8 + x9
                F2=~  + x4 + x5 + x6
                F3=~  + x1 + x2 + x3
                h  ~~ 0*F1
                h  ~~ 0*F2
                h  ~~ 0*F3
                F1 ~~ 0*F2
                F2 ~~ 0*F3
                F1 ~~ 0*F3


'
```

Once we have specified our model, we can proceed to tell `lavaan` to estimate the model. To fit the CFA model we will use `sem` function which has been harmonised with the functions `cfa` and `lavaan`. Remember that the kind of variables we have are categorical -so we include the option `ordered()` and we will request standardised loadings with `std.lv=TRUE`. We will store the output on the object `fit`.

```
fit <- sem(MD_model, data = Rel_MD_1,
           ordered=c("x1","x2","x3","x4","x5",
                     "x6","x7","x8","x9"),
           std.lv=TRUE)
# The command below is to check the output (We will check this in the
#next section and validity chapter)
```

```
#summary(fit, fit.measures=TRUE, rsquare=TRUE, standardized=TRUE)
```

Before showing the authomated way to estimate ω and ωh, we will show the manual computation. There are two main parameters one needs for their computation: factor loadings from the indicators to the overall factor (λh), to each dimension (λj) and the error. This can be easily extracted from the fit object as follows:

```
lambdas<-as.data.frame(fit@Model@GLIST$lambda)
error<-colSums(fit@Model@GLIST$theta)
```

The then the square of the sum of the loadings (λh) and (λj) is taken as well as the sum of the error. The we can compute both ω and ωh using equation @ref(eq:omega) and @ref(eq:omegah).

```
Slambda_2<-sum(lambdas[1])^2 + sum(lambdas[2])^2 +
           sum(lambdas[3])^2 + sum(lambdas[4])^2
error <- sum(error)

omega_t <- Slambda_2 / (Slambda_2+error)
omega_h <- sum(lambdas[1])^2 / (Slambda_2+error)
omegamanual<-c(omega_h=omega_h,omega_t=omega_t)
omegamanual
##   omega_h   omega_t
## 0.8445022 0.9707344
```

Fortunately, there is an R function from the `psych` package that does this for us. Once the model has been fitted, we apply the function `omegaFromSem()` to request the estimates and store the estimates of both ω and ωh in the `omegasem` object. The results indicate high overall reliability and high reliability after considering the multidimensional features of the scale.

```
omegasem<-omegaFromSem(fit)
omegasem<-c(omega_h=omegasem$omega,
            omega_t=omegasem$omega.tot)
omegasem
##   omega_h   omega_t
## 0.8446990 0.9707276
```

Item-level reliability

Estimation with R

Overall reliability is an excellent summary of the quality of an index in that it tells the homogeneity of our scale and its capacity to produce consistent population rankings. In section @ref(#relestimation) we showed that including some uncorrelated items lead to a reduction of reliability and that this has negative implication for consistency across measurements. This section focuses on item reliability, i.e. how specific items contribute positively or negatively to reliability.

Section @ref(itemlevelrel) reviewed Item Response Theory (IRT), which is a theory of the properties of indicators by putting forward the concepts of discrimination and severity. The standard IRT modelling assumes that scales are unidimensional in that indicators are manifest of a latent trait. However, in parallel to the development of CFA, IRT modelling has incorporated multidimensional models. However, as long as a scale is homogeneous, the bias from a multidimensional IRT and the unidimensional one should not dramatically change our conclusions about the reliability of the items.

To illustrate IRT modelling, we will work with the same data set, which we know that results in a highly homogeneous scale. We will use the data “Rel_MD_1” and the R-package `ltm` which fits different kinds of IRT models. The `ltm()` function fits one, two and three-parameter IRT models. Below we fit a two-parameter IRT model simply by adding the `z1` option so we allow the model to have different slopes, i.e. Two-parameter IRT model. The output shows the difficulty and discrimination coefficients. The difficulty represents the severity of the deprivation on the latent trait. For example, item x1 is less severe than item x3. The second column `Dscrmn` displays the values of the discrimination parameters. All the items have values $>.9$ which is above the suggested threshold by Guio et al. (2016) and Nájera (2018).

```
library(ltm)
rel_irt<-ltm(Rel_MD_1[,c(1:9)] ~ z1)
rel_irt
##
## Call:
## ltm(formula = Rel_MD_1[, c(1:9)] ~ z1)
##
## Coefficients:
##      Dffclt  Dscrmn
## x1    0.023   2.290
## x2    0.702   2.183
## x3    1.258   2.198
## x4    0.053   2.306
## x5    0.698   2.368
## x6    1.297   2.154
## x7    0.160   2.541
## x8    0.802   2.547
## x9    1.236   2.477
##
## Log.Lik: -20132.92
```

Before checking the estimation with Mplus, we will check what will happen if we include our unreliable items. Items x1 and x2 are replaced by items V1 and V2. As expected, the discrimination values are unacceptably low as well as the severity values which are (≥ 3) standard deviations.

```
head(Rel_MD_1[,c(3:11)])
##   x3 x4 x5 x6 x7 x8 x9 x10 x11
## 1  1  1  0  0  0  0  0   0   0
## 2  0  0  0  0  0  0  0   0   0
## 3  0  1  0  0  0  0  0   0   0
## 4  0  0  0  0  1  0  0   0   0
## 5  0  0  0  0  0  0  0   1   1
## 6  0  0  0  0  0  0  0   0   0
rel_irt_2<-ltm(Rel_MD_1[,c(3:11)] ~ z1)
rel_irt_2
##
## Call:
## ltm(formula = Rel_MD_1[, c(3:11)] ~ z1)
##
## Coefficients:
##      Dffclt  Dscrmn
## x3    1.426   1.651
## x4    0.053   2.371
## x5    0.689   2.488
## x6    1.279   2.245
## x7    0.157   2.817
## x8    0.780   2.840
## x9    1.193   2.804
## x10   5.142   0.133
## x11   4.882   0.135
##
## Log.Lik: -21740.2
```

Guio, Anne-Catherine, Eric Marlier, David Gordon, Eldin Fahmy, Shailen Nandy, and Marco Pomati. 2016. “Improving the Measurement of Material Deprivation at the European Union Level.” *Journal of European Social Policy* 26 (3): 219–333. https://doi.org/10.1177/0958928716642947.

Nájera, Héctor E. 2018. “Reliability, Population Classification and Weighting in Multidimensional Poverty Measurement: A Monte Carlo Study.” *Social Indicators Research*, June. https://doi.org/10.1007/s11205-018-1950-z.

Revelle, W. 2014. “Psych. R Package.” R software.

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

Schmid, J., and J. N.. Leiman. 1957. “The Development of Hierarchical Factor Solutions.” *Psychometrika* 22 (1): 53–61.