

Chapter 6: MIC package demo

R-package to estimate the adjusted MIC, predicted MIC and ROC-based MIC

The R-package is public and can be installed from GitHub: `remotes::install_github(repo = "iriseekhout/MIC")`.

```
# to install MIC package from Github:  
#remotes::install_github(repo = "iriseekhout/MIC")  
  
# Acquire packages  
library(mirt)      # for simulating data  
library(MIC)       # for calculating the MIC
```

Simulate dataset using ‘mirt’

This code simulates an example data set with 10 items that are measured at two time points. This dataset is used as an illustration.

Generate a set of IRT item parameters

```
set.seed(12345)  
  
b2 <- c(-0.8, -0.8, -0.4, -0.4, 0, 0, 0.4, 0.4, 0.8, 0.8)  
bc <- b2/4  
b1 <- b2 - 1 + sample(bc)  
b3 <- b2 + 1 + sample(bc)  
a1 <- sample( 1+b2/2 )  
  
( cf.simb <- data.frame(a1,b1,b2,b3) )
```

```
##      a1    b1    b2          b3  
## 1  1.4 -1.9 -0.8  4.000000e-01  
## 2  0.8 -1.7 -0.8 -5.551115e-17  
## 3  0.6 -1.6 -0.4  7.000000e-01  
## 4  1.2 -1.4 -0.4  7.000000e-01  
## 5  1.0 -0.8  0.0  1.000000e+00  
## 6  1.2 -0.8  0.0  1.200000e+00  
## 7  1.4 -0.5  0.4  1.300000e+00  
## 8  0.8 -0.6  0.4  1.200000e+00  
## 9  1.0 -0.3  0.8  1.700000e+00  
## 10 0.6 -0.4  0.8  1.800000e+00
```

Transform b-parameters to d-parameters (mirt works with d-parameters)

difficulty (b) = easiness (d) / -a

```
cf.sim <- cf.simb  
  
colnames(cf.sim) <- c("a1", "d1", "d2", "d3")  
  
cf.sim$d1 <- -cf.simb$b1*cf.sim$a1  
cf.sim$d2 <- -cf.simb$b2*cf.sim$a1  
cf.sim$d3 <- -cf.simb$b3*cf.sim$a1  
round(cf.sim, 3)
```

```
##      a1    d1    d2    d3  
## 1  1.4 2.66  1.12 -0.56  
## 2  0.8 1.36  0.64  0.00  
## 3  0.6 0.96  0.24 -0.42  
## 4  1.2 1.68  0.48 -0.84  
## 5  1.0 0.80  0.00 -1.00  
## 6  1.2 0.96  0.00 -1.44  
## 7  1.4 0.70 -0.56 -1.82  
## 8  0.8 0.48 -0.32 -0.96  
## 9  1.0 0.30 -0.80 -1.70  
## 10 0.6 0.24 -0.48 -1.08
```

```
round(colMeans(cf.sim), 3)
```

```
##      a1      d1      d2      d3  
## 1.000 1.014 0.032 -0.982
```

Create theta T1 and dataset 1 (baseline)

```
a1 <- as.matrix(cf.sim[, 1])  
d1 <- as.matrix(cf.sim[, -1])  
  
N <- 1000      # sample size  
prop.imp <- 0.3 # proportion improved  
mn.imic <- 0.5 # mean individual MICs in terms of theta change  
  
tet1s <- rnorm(N, 0, 1) # simulate theta T1  
tet1s <- as.matrix(tet1s)  
  
set.seed( sample(10000:20000,1) )  
  
dat1 <- simdata(a1, d1, N, itemtype="graded", Theta=tet1s)  
dat1 <- as.data.frame(dat1)  
  
xo1 <- rowSums(dat1) # This is the baseline test (sum) score
```

Create theta change

```
rt1ch <- -0.5          # correlation between theta T1 and theta change
tetchs <- rt1ch*tet1s + sqrt(1-rt1ch^2)*rnorm(N, mean(tet1s), sd(tet1s))
cor(tet1s,tetchs)
```

```
##           [,1]
## [1,] -0.5012465
```

```
tetchs <- tetchs - mean(tetchs)      # makes mean = 0
tetchs <- (tetchs/sd(tetchs))*1      # makes SD = 1
```

```
( qtl <- quantile(tetchs, prob=(1-prop.imp)) )
```

```
##           70%
## 0.4724834
```

```
( mean.tetchs <- mn.imic - qtl )      # Estimate mean theta change to get
```

```
##           70%
## 0.02751659
```

```
# the desired proportion improved
tetchs <- tetchs + mean.tetchs      # transform mean of tetchs
```

Create theta T2 and dataset 2 (follow-up)

```
tet2s <- as.matrix(tet1s + tetchs)  # theta T2
```

```
cor(tet1s, tet2s)
```

```
##           [,1]
## [1,] 0.4855813
```

```
set.seed( sample(30000:40000,1) )
```

```
dat2 <- simdata(a1, d1, N, itemtype="graded", Theta=tet2s)
dat2 <- as.data.frame(dat2)
```

Insert error in transition ratings

```
rel.trat <- 0.5      # reliability of the perceived change/transition rating
```

```
( sd.ch.error <- sqrt(((1-rel.trat)/rel.trat)*sd(tetchs)^2) )
```

```
## [1] 1
```

```
tetch.percv <- tetchs + rnorm(N, 0, sd.ch.error) # perceived change
```

Create iMIC distribution

```
imic <- rnorm(N, mn.imic, 0.1*mn.imic)
```

create dichotomous transition ratings

```
trat <- numeric(N)
trat[tetch.percv > imic] <- 1
mean(trat) # proportion improved based on perceived change
```

```
## [1] 0.355
```

Create dataset

```
org <- data.frame(dat1, dat2, trat)
head(org)
```

```
##   Item_1 Item_2 Item_3 Item_4 Item_5 Item_6 Item_7 Item_8 Item_9 Item_10
## 1      1      0      1      0      0      0      0      0      0      2
## 2      3      2      0      2      2      0      1      1      3      0
## 3      3      0      0      3      0      3      0      0      2      0
## 4      2      0      0      0      2      1      0      0      0      3
## 5      3      2      2      1      0      3      1      1      0      1
## 6      3      3      2      0      0      0      0      3      0      0
##   Item_1.1 Item_2.1 Item_3.1 Item_4.1 Item_5.1 Item_6.1 Item_7.1 Item_8.1
## 1      1      3      1      2      0      0      0      2
## 2      2      0      2      2      2      2      2      1
## 3      3      3      3      3      3      3      3      2
## 4      0      3      1      2      2      1      2      3
## 5      3      1      3      3      3      0      2      3
## 6      3      0      3      0      0      1      3      0
##   Item_9.1 Item_10.1 trat
## 1      0      0      1
## 2      0      0      0
## 3      2      2      1
## 4      0      1      0
## 5      2      3      1
## 6      0      0      0
```

```
nitems = 10 # Provide the number of items in the scale
```

Simplify/standardize the item names

```
names(org)[1:nitems] <- paste0('v1', '_', 1:nitems)
names(org)[(nitems+1):(2*nitems)] <- paste0('v2', '_', 1:nitems)
names(org)[2*nitems+1] <- "trat"
names(org)
```

```
## [1] "v1_1" "v1_2" "v1_3" "v1_4" "v1_5" "v1_6" "v1_7" "v1_8" "v1_9"
## [10] "v1_10" "v2_1" "v2_2" "v2_3" "v2_4" "v2_5" "v2_6" "v2_7" "v2_8"
## [19] "v2_9" "v2_10" "trat"
```

```
## copy original data into a workfile (dat)
```

```
dat <- org
```

```
head(dat)
```

```
##   v1_1 v1_2 v1_3 v1_4 v1_5 v1_6 v1_7 v1_8 v1_9 v1_10 v2_1 v2_2 v2_3 v2_4 v2_5
## 1    1    0    1    0    0    0    0    0    0    2    1    3    1    2    0
## 2    3    2    0    2    2    0    1    1    3    0    2    0    2    2    2
## 3    3    0    0    3    0    3    0    0    2    0    3    3    3    3    3
## 4    2    0    0    0    2    1    0    0    0    3    0    3    1    2    2
## 5    3    2    2    1    0    3    1    1    0    1    3    1    3    3    3
## 6    3    3    2    0    0    0    0    3    0    0    3    0    3    0    0
##   v2_6 v2_7 v2_8 v2_9 v2_10 trat
## 1    0    0    2    0    0    1
## 2    2    2    1    0    0    0
## 3    3    3    2    2    2    1
## 4    1    2    3    0    1    0
## 5    0    2    3    2    3    1
## 6    1    3    0    0    0    0
```

Estimate TR reliability with lavaan

Specify the factor model

A factor for each time point, with correlated errors over time.

```
model <- '
  # Factors
  F1 =~ v1_1+v1_2+v1_3+v1_4+v1_5+v1_6+v1_7+v1_8+
        v1_9+v1_10+trat
  F2 =~ v2_1+v2_2+v2_3+v2_4+v2_5+v2_6+v2_7+v2_8+
        v2_9+v2_10+trat

  # Correlated errors over time
  v1_1 ~~ v2_1
  v1_2 ~~ v2_2
  v1_3 ~~ v2_3
  v1_4 ~~ v2_4
  v1_5 ~~ v2_5
  v1_6 ~~ v2_6
  v1_7 ~~ v2_7
  v1_8 ~~ v2_8
  v1_9 ~~ v2_9
  v1_10 ~~ v2_10
  '
```

Use the `tr_reliability` function

```
tr_reliability(example, model)
```

```
## $reliability
## [1] 0.4916676
##
## $'standardized modification index for lv'
## [1] lhs      op      rhs      mi      epc      sepc.lv  sepc.all sepc.nox
## <0 rows> (or 0-length row.names)
```

Note: Check the model fit. If necessary improve model fit by allowing correlated errors cross-sectionally (e.g., $v1_1 \sim v1_2$, $v2_1 \sim v2_2$)

```
reliability <- tr_reliability(example, model, modification = FALSE)
```

Reliability of TR

Estimate MICs

First prepare the sumscores in the data, to calculate the minimal important change for.

```
nitems <- 10
example$x <- rowSums(example[,1:nitems])      # sumscore T1
example$y <- rowSums(example[, (nitems+1):(2*nitems)]) # sumscore T2
```

ROC-based MIC

```
mic_roc(example, x = "x", y = "y", tr = "trat")
```

```
## ROC-based MIC
##           -0.5
```

predicted MIC

```
mic_pred(example, x = "x", y = "y", tr = "trat")
```

```
## predicted MIC
##           0.8484085
```

Adjusted MIC

```
mic_adjust(example, x = "x", y = "y", tr = "trat", reliability = reliability)
```

```
## adjusted MIC  
##      2.80477
```

Bootstrap confidence intervals

ROC-based MIC

```
start <- Sys.time()  
bootCImic(example, mic = "roc", x = "x", y = "y", tr = "trat")
```

```
## # A tibble: 1 x 6  
##   term      .lower .estimate .upper .alpha .method  
##   <chr>    <dbl>    <dbl> <dbl> <dbl> <chr>  
## 1 mic_roc  -2.5    -0.393    2.5  0.05 percentile
```

```
Sys.time() - start
```

```
## Time difference of 7.053701 secs
```

predicted MIC

```
start <- Sys.time()  
bootCImic(example, mic = "predict", x = "x", y = "y", tr = "trat")
```

```
## # A tibble: 1 x 6  
##   term      .lower .estimate .upper .alpha .method  
##   <chr>    <dbl>    <dbl> <dbl> <dbl> <chr>  
## 1 mic_pred 0.407    0.847    1.27  0.05 percentile
```

```
Sys.time() - start
```

```
## Time difference of 6.451905 secs
```

adjusted MIC

```
start <- Sys.time()  
bootCImic(example, mic = "adjust", x = "x", y = "y", tr = "trat", model = model)
```

```
## # A tibble: 1 x 6  
##   term      .lower .estimate .upper .alpha .method  
##   <chr>    <dbl>    <dbl> <dbl> <dbl> <chr>  
## 1 mic_adjust 1.97    2.80    3.86  0.05 percentile
```

```
Sys.time() - start
```

```
## Time difference of 16.94754 mins
```