Early-stopping CAT simulations

Piotr Król

09 08 2021

library(ltm)  
library(mirt)  
library(mirtCAT)  
library(parallel)  
library(plyr)  
library(ggpubr)

responses\_demo <- read.csv("Data/responses\_demo.csv", encoding = "UTF-8")  
responses <- as.matrix(responses\_demo[,5:ncol(responses\_demo)])  
load("Data/mod\_r")  
load("Data/items\_to\_remove")  
responses <- responses[, -items\_to\_remove]

mod <- mod\_r  
mod

##   
## Call:  
## mirt(data = responses[, -c(392, 417, 652)], model = 1, SE = TRUE,   
## quadpts = 601, technical = list(NCYCLES = 10000))  
##   
## Full-information item factor analysis with 1 factor(s).  
## Converged within 1e-04 tolerance after 2700 EM iterations.  
## mirt version: 1.34   
## M-step optimizer: BFGS   
## EM acceleration: Ramsay   
## Number of rectangular quadrature: 601  
## Latent density type: Gaussian   
##   
## Information matrix estimated with method: Oakes  
## Second-order test: model is a possible local maximum  
## Condition number of information matrix = 2311.374  
##   
## Log-likelihood = -460055  
## Estimated parameters: 1334   
## AIC = 922778.1; AICc = 926480.5  
## BIC = 930434.4; SABIC = 926196  
## G2 (9999998666) = 885020.2, p = 1  
## RMSEA = 0, CFI = NaN, TLI = NaN

Calculate Cronbach’s alpha for responses:

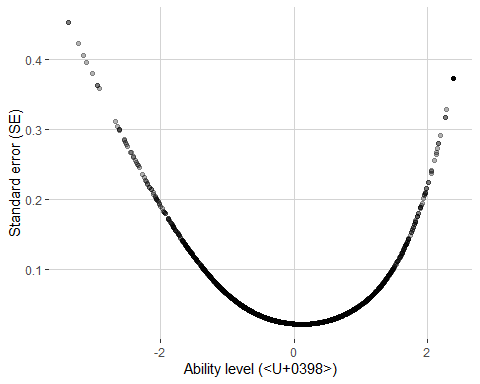
cronbach.alpha(responses)

##   
## Cronbach's alpha for the 'responses' data-set  
##   
## Items: 667  
## Sample units: 2297  
## alpha: 0.999

Prepare full scores, params and mirt object for simulations:

fscores <- as.data.frame(fscores(mod, method = "MAP", full.scores.SE = T)) #Obtain full scores  
params <- as.data.frame(coef(mod, simplify = T)$items)[1:2] #Prepare params  
mo <- generate.mirt\_object(params, '2PL') #Prepare mirt object

ggplot(fscores, aes(F1, SE\_F1)) +  
 geom\_point(alpha = 0.3) +  
 xlab("Ability level (Θ)") +  
 ylab("Standard error (SE)") +  
 theme(  
 panel.background = element\_rect(fill = NA),  
 panel.grid.major = element\_line(colour = "lightgrey")  
)



summary(fscores$SE\_F1)

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 0.02181 0.02500 0.03427 0.06199 0.06391 0.45343

Make early-stopping simulation with desired SE = 0.1:

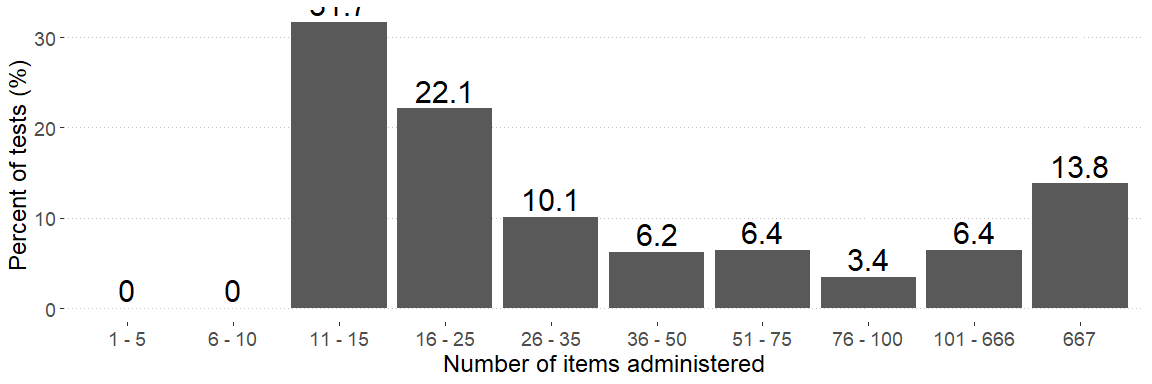
#Prepare cluster for faster simulation  
cl <- makeCluster(detectCores())  
  
#Make simulation  
results\_SE1 <- mirtCAT(mo = mo, method = "MAP", criteria = "MI", start\_item = "MI", local\_pattern = responses, cl = cl, design = list(min\_SEM = 0.1))  
save.image("Data/early\_stop\_CAT.RData")

sim\_results <- results\_SE1  
  
#Obtain mean test length  
tests\_lengths <- laply(sim\_results, function(x) length(x$items\_answered))  
mean\_length <- round(mean(tests\_lengths), 1)  
  
#Obtain median test length  
median\_length <- round(median(tests\_lengths), 1)  
  
#Obtain thetas  
thetas <- laply(sim\_results, function(x) x$thetas)  
  
#Get correlation of thetas with full scores  
cor <- round(cor(thetas, fscores$F1), 3)  
  
#Get mean SE  
meanSE <- round(mean(laply(sim\_results, function(x) x$SE\_thetas)), 3)  
  
#Get reliability  
rel <- round(1 - meanSE\*\*2, 3)  
  
#Get number of unused items  
raw\_responses <- laply(sim\_results, function(x) x$raw\_responses)  
items\_used <- length(which(apply(raw\_responses, 2, function(x) any(!is.na(x)))))  
unused <- nrow(params) - items\_used  
  
paste("Mean length:", mean\_length, " Median length:", median\_length, " Correlation:", cor, " Mean SE:", meanSE, " Reliability:", rel, " Unused:", unused)

## [1] "Mean length: 126.4 Median length: 23 Correlation: 0.99 Mean SE: 0.113 Reliability: 0.987 Unused: 0"

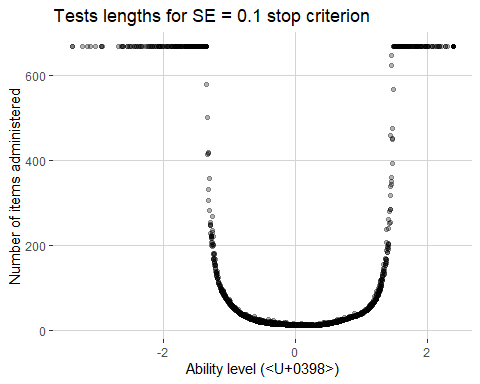
Plot distribution of tests lengths:

#Prepare cuts  
cuts <- cut(tests\_lengths, breaks = c(0, 5, 10, 15, 25, 35, 50, 75, 100, 666, 667), labels = c("1 - 5", "6 - 10", "11 - 15", "16 - 25", "26 - 35", "36 - 50", "51 - 75", "76 - 100", "101 - 666", "667"))  
  
#Plot  
ggplot(data.frame(round(table(cuts) / nrow(responses) \* 100, 1)), aes(x = cuts, y = Freq)) +  
 xlab("Number of items administered") +  
 ylab("Percent of tests (%)") +  
 geom\_bar(stat = "identity") +  
 geom\_text(aes(label = Freq), vjust = -0.3, size=8) +   
 theme\_pubclean() +  
 theme(text = element\_text(size=18))



Plot theta vs. test length:

ggplot(data.frame(thetas = thetas, lengths = tests\_lengths), aes(thetas, lengths)) +  
 geom\_point(alpha = 0.3) +  
 xlab("Ability level (Θ)") +  
 ylab("Number of items administered") +  
 ggtitle("Tests lengths for SE = 0.1 stop criterion") +  
 theme(  
 panel.background = element\_rect(fill = NA),  
 panel.grid.major = element\_line(colour = "lightgrey")  
)



Make early-stopping simulation with different maximum number of items administered:

lengths <- c(5, 10, 15, 25, 35, 50, 75, 100)  
results\_SEXX <- data.frame(mean\_length = NA, median\_length = NA, cor = NA, meanSE = NA, rel = NA, unused = NA)  
cl <- makeCluster(detectCores()) #Speeds up simulations - computation in parallel  
  
for (n in lengths){  
   
 #Make simulation  
 sim\_results <- mirtCAT(mo = mo, method = "MAP", criteria = "MI", start\_item = "MI", local\_pattern = responses, cl = cl, design = list(min\_SEM = XX, max\_items = n))  
   
 #Obtain mean test length  
 tests\_lengths <- laply(sim\_results, function(x) length(x$items\_answered))  
 mean\_length <- round(mean(tests\_lengths), 1)  
   
 #Obtain median test length  
 median\_length <- round(median(tests\_lengths), 1)  
   
 #Obtain thetas  
 thetas <- laply(sim\_results, function(x) x$thetas)  
   
 #Get correlation of thetas with full scores  
 cor <- round(cor(thetas, fscores$F1), 3)  
   
 #Get mean SE  
 meanSE <- round(mean(laply(sim\_results, function(x) x$SE\_thetas)), 3)  
   
 #Get reliability  
 rel <- round(1 - meanSE\*\*2, 3)  
   
 #Get number of unused items  
 raw\_responses <- laply(sim\_results, function(x) x$raw\_responses)  
 items\_used <- length(which(apply(raw\_responses, 2, function(x) any(!is.na(x)))))  
 unused <- nrow(params) - items\_used  
   
 #Update data frame with results  
 results\_SEXX <- rbind(results\_SEXX, c(mean\_length, median\_length, cor, meanSE, rel, unused))  
   
}  
  
results\_SEXX <- na.omit(results\_SEXX)  
row.names(results\_SEXX) <- lengths  
save.image("Data/early\_stop\_CAT.RData")

#results\_fixed\_length