**Appendix A:**

**Code for multiple forms using lpsolveAPI**

# Load packages and data ---------------------------------------------------

library(readxl)

library(lpSolveAPI)

library(dplyr)

Bank\_Collection <- read\_excel("Demo data.xlsx", sheet = "Metadata")

enemies <- read\_excel("Demo data.xlsx", sheet = "Enemies")

enemies <- subset(enemies, select= -c(TYPE))

Bank\_Collection$Content <- substr(Bank\_Collection$Comp, 3, 5)

Bank\_Collection$Content[Bank\_Collection$Content=="1.1"] <- 1

Bank\_Collection$Content[Bank\_Collection$Content=="1.2"] <- 2

Bank\_Collection$Content[Bank\_Collection$Content=="1.3"] <- 3

Bank\_Collection$Content[Bank\_Collection$Content=="1.4"] <- 4

Bank\_Collection$Content[Bank\_Collection$Content=="1.5"] <- 5

Bank\_Collection$Content[Bank\_Collection$Content=="2.1"] <- 6

Bank\_Collection$Content[Bank\_Collection$Content=="2.2"] <- 7

Bank\_Collection$Content[Bank\_Collection$Content=="2.3"] <- 8

Bank\_Collection$Content <- as.numeric(Bank\_Collection$Content)

Bank\_Collection$B <- as.numeric(Bank\_Collection$IRTb)

Bank\_Collection$ID <- as.numeric(Bank\_Collection$ID)

OP\_items <- subset(Bank\_Collection, Status == "Scored Items (Sco)")

OP\_items$seq <- seq(1:nrow(OP\_items))

attach(OP\_items)

attach(enemies)

# Define Variables --------------------------------------------------------

n\_items <- nrow(OP\_items) #Number of items in pool (I)

n\_forms <- 10 #Number of test forms (F)

tif\_points <- 1 #Number of theta points at which information if controlled (J)

form\_length <- 100 #Length of both test forms (N)

n\_content <- c(10, 50, 11, 1, 6, 10, 5, 7) #Numbers of items required from k=1:8 content categories (Nc)

# Set TIF parameters ------------------------------------------------------

theta <- c(-0.33) #Define I? points at which information is controlled

Info <- array(0,c(n\_items,tif\_points)) #Define empty matrix with item information function values

#Fill matrix with item information function values (Rasch model)

for(j in 1:tif\_points){

P <-exp(theta[j]-B)/(1+exp(theta[j]-B))

Q <- 1-P

Info[,j] <- P\*Q

}

# Content Categories ------------------------------------------------------

Vc <- list()

for(k in 1:length(n\_content)){

Vc[[k]] <- c(1:n\_items)[Content==k]

}

# Enemy Items -------------------------------------------------------------

enemies$itemA.index<-match(enemies$ItemA, OP\_items$ID)

enemies$itemB.index<-match(enemies$ItemB, OP\_items$ID)

Ve <- list()

library(purrr)

Ve<-enemies[,3:4] %>% purrr::transpose()

Ve<-lapply(Ve, unlist, use.names=FALSE)

Ve<-unique(lapply(Ve, sort))

Ve<- Filter(function(x) length(x)>1, Ve)

# Set Linear Programming Parameters ---------------------------------------

start\_time <- Sys.time()

# Number of decision variables number of items by the number of forms

n\_dec\_vars <- n\_items\*n\_forms

# Create empty model

lprec <- make.lp(0,n\_dec\_vars)

# Objective function

set.objfn(lprec,rep(Info,n\_forms))

# Set Constraints ---------------------------------------------------------

# Constraints in Eq. 5 (Ensures no overlap in tests)

for (k in 1:n\_items){

add.constraint(lprec,rep(1,10),"<=",1,indices=c(k,

n\_items+k,

2\*n\_items+k,

3\*n\_items+k,

4\*n\_items+k,

5\*n\_items+k,

6\*n\_items+k,

7\*n\_items+k,

8\*n\_items+k,

9\*n\_items+k)

)

}

# Constraints in Eq. 6 (Set content categories Form 1)

for (k in 1:length(n\_content)){

add.constraint(lprec,rep(1,length(Vc[[k]])),">=",n\_content[k],indices=Vc[[k]])

}

# Constraints in Eq. 6 (Set content categories Form 2)

for (k in 1:length(n\_content)){

add.constraint(lprec,rep(1,length(Vc[[k]])),">=",n\_content[k],indices=n\_items+Vc[[k]])

}

# Constraints in Eq. 6 (Set content categories Form 3)

for (k in 1:length(n\_content)){

add.constraint(lprec,rep(1,length(Vc[[k]])),">=",n\_content[k],indices=2\*n\_items+Vc[[k]])

}

# Constraints in Eq. 6 (Set content categories Form 4)

for (k in 1:length(n\_content)){

add.constraint(lprec,rep(1,length(Vc[[k]])),">=",n\_content[k],indices=3\*n\_items+Vc[[k]])

}

# Constraints in Eq. 6 (Set content categories Form 5)

for (k in 1:length(n\_content)){

add.constraint(lprec,rep(1,length(Vc[[k]])),">=",n\_content[k],indices=4\*n\_items+Vc[[k]])

}

# Constraints in Eq. 6 (Set content categories Form 6)

for (k in 1:length(n\_content)){

add.constraint(lprec,rep(1,length(Vc[[k]])),">=",n\_content[k],indices=5\*n\_items+Vc[[k]])

}

# Constraints in Eq. 6 (Set content categories Form 7)

for (k in 1:length(n\_content)){

add.constraint(lprec,rep(1,length(Vc[[k]])),">=",n\_content[k],indices=6\*n\_items+Vc[[k]])

}

# Constraints in Eq. 6 (Set content categories Form 8)

for (k in 1:length(n\_content)){

add.constraint(lprec,rep(1,length(Vc[[k]])),">=",n\_content[k],indices=7\*n\_items+Vc[[k]])

}

# Constraints in Eq. 6 (Set content categories Form 9)

for (k in 1:length(n\_content)){

add.constraint(lprec,rep(1,length(Vc[[k]])),">=",n\_content[k],indices=8\*n\_items+Vc[[k]])

}

# Constraints in Eq. 6 (Set content categories Form 10)

for (k in 1:length(n\_content)){

add.constraint(lprec,rep(1,length(Vc[[k]])),">=",n\_content[k],indices=9\*n\_items+Vc[[k]])

}

# Constraints for enemy items

# Set enemies Form 1

for (e in 1:length(Ve)){

add.constraint(lprec,rep(1,length(Ve[[e]])),"<=",1,indices=Ve[[e]])

}

# Set enemies Form 2

for (e in 1:length(Ve)){

add.constraint(lprec,rep(1,length(Ve[[e]])),"<=",1,indices=n\_items+Ve[[e]])

}

# Set enemies Form 3

for (e in 1:length(Ve)){

add.constraint(lprec,rep(1,length(Ve[[e]])),"<=",1,indices=2\*n\_items+Ve[[e]])

}

# Set enemies Form 4

for (e in 1:length(Ve)){

add.constraint(lprec,rep(1,length(Ve[[e]])),"<=",1,indices=3\*n\_items+Ve[[e]])

}

# Set enemies Form 5

for (e in 1:length(Ve)){

add.constraint(lprec,rep(1,length(Ve[[e]])),"<=",1,indices=4\*n\_items+Ve[[e]])

}

# Set enemies Form 6

for (e in 1:length(Ve)){

add.constraint(lprec,rep(1,length(Ve[[e]])),"<=",1,indices=5\*n\_items+Ve[[e]])

}

# Set enemies Form 7

for (e in 1:length(Ve)){

add.constraint(lprec,rep(1,length(Ve[[e]])),"<=",1,indices=6\*n\_items+Ve[[e]])

}

# Set enemies Form 8

for (e in 1:length(Ve)){

add.constraint(lprec,rep(1,length(Ve[[e]])),"<=",1,indices=7\*n\_items+Ve[[e]])

}

# Set enemies Form 9

for (e in 1:length(Ve)){

add.constraint(lprec,rep(1,length(Ve[[e]])),"<=",1,indices=8\*n\_items+Ve[[e]])

}

# Set enemies Form 10

for (e in 1:length(Ve)){

add.constraint(lprec,rep(1,length(Ve[[e]])),"<=",1,indices=9\*n\_items+Ve[[e]])

}

# Constraints in Eq. 7 (Sets the number of items in each form)

add.constraint(lprec, rep(1,n\_items), "=", form\_length, indices = 1:n\_items)

add.constraint(lprec, rep(1,n\_items), "=", form\_length, indices = (n\_items+1):(2\*n\_items))

add.constraint(lprec, rep(1,n\_items), "=", form\_length, indices = (2\*n\_items+1):(3\*n\_items))

add.constraint(lprec, rep(1,n\_items), "=", form\_length, indices = (3\*n\_items+1):(4\*n\_items))

add.constraint(lprec, rep(1,n\_items), "=", form\_length, indices = (4\*n\_items+1):(5\*n\_items))

add.constraint(lprec, rep(1,n\_items), "=", form\_length, indices = (5\*n\_items+1):(6\*n\_items))

add.constraint(lprec, rep(1,n\_items), "=", form\_length, indices = (6\*n\_items+1):(7\*n\_items))

add.constraint(lprec, rep(1,n\_items), "=", form\_length, indices = (7\*n\_items+1):(8\*n\_items))

add.constraint(lprec, rep(1,n\_items), "=", form\_length, indices = (8\*n\_items+1):(9\*n\_items))

add.constraint(lprec, rep(1,n\_items), "=", form\_length, indices = (9\*n\_items+1):(10\*n\_items))

# Constraints in Eqs. 8-9

set.type(lprec,columns=c(1:(n\_forms\*n\_items)),type="binary")

set.type(lprec,columns=n\_dec\_vars,type="real")

set.bounds(lprec,lower=rep(0,n\_dec\_vars),upper=rep(1,n\_dec\_vars))

# SOLVE ------------------------------------------------------------------

# set to maximize the objective function at the desired point

lp.control(lprec,sense="max");

#Solve model

solve(lprec)

# Get the value of the objective function

get.objective(lprec)

# Get the values of the decision variables

get.variables(lprec)

# Get realizations of the constraints

get.constraints(lprec)

end\_time <- Sys.time()

end\_time - start\_time

# Plot Test Information Functions -----------------------------------------

decvar <- get.variables(lprec) #Insert values for decision variables in new vector decvar

x <- seq(-3,3, .01) #Define theta axis

ItemInfo <- array(0,c(n\_items, length(x))) #Define empty matrix with item information function values along theta axis for all items

for (j in 1:length(x)){

P <-exp(x[j]-B)/(1+exp(x[j]-B))

Q <- 1-P

ItemInfo[,j] <- P\*Q

} #Calculate item information function values along theta axis for all items

y = matrix(0,length(x),10) #Define empty matrix with item information function values along theta axis for all items

for (j in 1:length(x)){

#Calculate test information function values along theta axis for Form 1

y[j,1]=sum(decvar[1:n\_items]\*ItemInfo[,j])

#Calculate test information function values along theta axis for Form 2

y[j,2]=sum(decvar[(n\_items+1):(2\*n\_items)]\*ItemInfo[,j])

#Calculate test information function values along theta axis for Form 3

y[j,3]=sum(decvar[(2\*n\_items+1):(3\*n\_items)]\*ItemInfo[,j])

#Calculate test information function values along theta axis for Form 4

y[j,4]=sum(decvar[(3\*n\_items+1):(4\*n\_items)]\*ItemInfo[,j])

#Calculate test information function values along theta axis for Form 5

y[j,5]=sum(decvar[(4\*n\_items+1):(5\*n\_items)]\*ItemInfo[,j])

#Calculate test information function values along theta axis for Form 6

y[j,6]=sum(decvar[(5\*n\_items+1):(6\*n\_items)]\*ItemInfo[,j])

#Calculate test information function values along theta axis for Form 7

y[j,7]=sum(decvar[(6\*n\_items+1):(7\*n\_items)]\*ItemInfo[,j])

#Calculate test information function values along theta axis for Form 8

y[j,8]=sum(decvar[(7\*n\_items+1):(8\*n\_items)]\*ItemInfo[,j])

#Calculate test information function values along theta axis for Form 9

y[j,9]=sum(decvar[(8\*n\_items+1):(9\*n\_items)]\*ItemInfo[,j])

#Calculate test information function values along theta axis for Form 10

y[j,10]=sum(decvar[(9\*n\_items+1):(10\*n\_items)]\*ItemInfo[,j])

}

#Plot information functions

#jpeg("TIF.jpg")

plot(x,y[,1], type="l", lty=1, xlab='Theta',ylab='Information', col="gray1")

lines(x,y[,2], type="l", lty=1, xlab='Theta',ylab='Information', col="gray2")

lines(x,y[,3], type="l", lty=1, xlab='Theta',ylab='Information', col="gray3")

lines(x,y[,4], type="l", lty=1, xlab='Theta',ylab='Information', col="gray4")

lines(x,y[,5], type="l", lty=1, xlab='Theta',ylab='Information', col="gray5")

lines(x,y[,6], type="l", lty=1, xlab='Theta',ylab='Information', col="gray6")

lines(x,y[,7], type="l", lty=1, xlab='Theta',ylab='Information', col="gray7")

lines(x,y[,8], type="l", lty=1, xlab='Theta',ylab='Information', col="gray8")

lines(x,y[,9], type="l", lty=1, xlab='Theta',ylab='Information', col="gray9")

lines(x,y[,10], type="l", lty=1, xlab='Theta',ylab='Information', col="gray10")

abline(v = theta, col="red", lwd=3, lty=2)

#title(i)

#dev.off()

# Assign Items to Forms ---------------------------------------------------

OP\_items$FORM\_1 <-decvar[1:n\_items]

OP\_items$FORM\_2 <-decvar[(n\_items+1):(2\*n\_items)]

OP\_items$FORM\_3 <-decvar[(2\*n\_items+1):(3\*n\_items)]

OP\_items$FORM\_4 <-decvar[(3\*n\_items+1):(4\*n\_items)]

OP\_items$FORM\_5 <-decvar[(4\*n\_items+1):(5\*n\_items)]

OP\_items$FORM\_6 <-decvar[(5\*n\_items+1):(6\*n\_items)]

OP\_items$FORM\_7 <-decvar[(6\*n\_items+1):(7\*n\_items)]

OP\_items$FORM\_8 <-decvar[(7\*n\_items+1):(8\*n\_items)]

OP\_items$FORM\_9 <-decvar[(8\*n\_items+1):(9\*n\_items)]

OP\_items$FORM\_10 <-decvar[(9\*n\_items+1):(10\*n\_items)]

# Create output tables ----------------------------------------------------

table.blueprint<-OP\_items %>%

group\_by(Content) %>%

summarize(f1=sum(FORM\_1),

f2=sum(FORM\_2),

f3=sum(FORM\_3),

f4=sum(FORM\_4),

f5=sum(FORM\_5),

f6=sum(FORM\_6),

f7=sum(FORM\_7),

f8=sum(FORM\_8),

f9=sum(FORM\_9),

f10=sum(FORM\_10)

)

form\_1\_array<-OP\_items$ID[OP\_items$FORM\_1==1]

form\_2\_array<-OP\_items$ID[OP\_items$FORM\_2==1]

form\_3\_array<-OP\_items$ID[OP\_items$FORM\_3==1]

form\_4\_array<-OP\_items$ID[OP\_items$FORM\_4==1]

form\_5\_array<-OP\_items$ID[OP\_items$FORM\_5==1]

form\_6\_array<-OP\_items$ID[OP\_items$FORM\_6==1]

form\_7\_array<-OP\_items$ID[OP\_items$FORM\_7==1]

form\_8\_array<-OP\_items$ID[OP\_items$FORM\_8==1]

form\_9\_array<-OP\_items$ID[OP\_items$FORM\_9==1]

form\_10\_array<-OP\_items$ID[OP\_items$FORM\_10==1]

# tables for enemy items

enemies<-enemies %>%

mutate (f1 = ifelse(ItemA %in% form\_1\_array & ItemB %in% form\_1\_array, 1, 0),

f2 = ifelse(ItemA %in% form\_2\_array & ItemB %in% form\_2\_array, 1, 0),

f3 = ifelse(ItemA %in% form\_3\_array & ItemB %in% form\_3\_array, 1, 0),

f4 = ifelse(ItemA %in% form\_4\_array & ItemB %in% form\_4\_array, 1, 0),

f5 = ifelse(ItemA %in% form\_5\_array & ItemB %in% form\_5\_array, 1, 0),

f6 = ifelse(ItemA %in% form\_6\_array & ItemB %in% form\_6\_array, 1, 0),

f7 = ifelse(ItemA %in% form\_7\_array & ItemB %in% form\_7\_array, 1, 0),

f8 = ifelse(ItemA %in% form\_8\_array & ItemB %in% form\_8\_array, 1, 0),

f9 = ifelse(ItemA %in% form\_9\_array & ItemB %in% form\_9\_array, 1, 0),

f10 = ifelse(ItemA %in% form\_10\_array & ItemB %in% form\_10\_array, 1, 0) )

# Table produces a 0/1 result if any enemy pairs are found in the form

table.enemies <- enemies %>%

summarize(f1 = max(f1),

f2 = max(f2),

f3 = max(f3),

f4 = max(f4),

f5 = max(f5),

f6 = max(f6),

f7 = max(f7),

f8 = max(f8),

f9 = max(f9),

f10 = max(f10)

)