

# The Computation of $\tau_1$ , $\tau_2$ , $\tau_1^*$ , and $\tau_2^*$

L. Andries van der Ark, Wilco H. M. Emons, Klaas Sijtsma

May 10, 2007

## 1 Introduction

Van der Ark, Emons, and Sijtsma (2007; abbreviated to AES) proposed four new answer-copying statistics:  $\tau_1$ ,  $\tau_2$ ,  $\tau_1^*$ , and  $\tau_2^*$ . The computation of these statistics is explained in this report, which can be viewed as an appendix to AES. For an introduction to the answer-copying statistics, justifications for using the statistics, statistical properties of the statistics, and the statistical notation used, we refer to AES.

References to equations from AES are indicated by an asterisk. For example, “Equation 1\*” refers to Equation 1 in AES, whereas “Equation 1” refers to Equation 1 in this report. Section 2 describes the computation of  $\tau_1$ , section 3 describes the computation of  $\tau_2$ , and section 4 describes the computation of  $\tau_1^*$  and  $\tau_2^*$ . Computer code for the programming language R (R Development Core Team, 2006) to carry out the computations are given in eleven figures. For the computer code it is assumed that the item responses are collected in an  $N \times J$  matrix  $\mathbf{Y}$ , the item scores are collected in an  $N \times J$  matrix  $\mathbf{X}$ , the test version of the respondents are collected in a vector  $\mathbf{v}$  of length  $N$ , and the status of the items are collected in a boolean vector  $\mathbf{u}$  of length  $J$  (if item  $j$  is a unique item then  $u_j := \text{TRUE}$ ; if item  $j$  is a common item then  $u_j := \text{FALSE}$ ). Matrices  $\mathbf{X}$  and  $\mathbf{Y}$  and vectors  $\mathbf{u}$  and  $\mathbf{v}$  that were used in AES are available from <http://spitswww.uvt.nl/avdrark/research/research.htm#OtherPublicationsMark>.

The data are obtained in R by saving the data file `AES.dmp` in a certain directory (e.g., `C:/datafiles/`) and executing the the following commands in R (modify the first command depending on the directory where the data were saved):

```
path <- "C:/datafiles/"
AESdata <- source(paste(path, "AES.dmp", sep=""))$value
```

Matrices  $\mathbf{X}$  and  $\mathbf{Y}$  and vectors  $\mathbf{u}$  and  $\mathbf{v}$  needed for computation of the answer0copying statistics are obtained by executing the following commands in R

```
Y <- AESdata[,2:25]
v <- AESdata[,1]
X <- rbind(as.matrix(AESdata[v==1,28:51]), as.matrix(AESdata[v==2,52:75]))
u <- c(F,F,F,F,F,T,T,T,T,T,T,F,T,T,T,T,F,T,T,T,T,F,T,T)
```

## 2 The computation of $\tau_1$ .

Equation 2\* shows that for examinee pair  $(v, w)$ ,  $\tau_1$  is computed as

$$\tau_{1vw} = \frac{T_{1vw} - E(T_{1vw})}{T_{1vw}^{\max} - E(T_{1vw})}. \quad (1)$$

Equation 1 contains three statistics:  $T_{1vw}$ ,  $E(T_{1vw})$ , and  $T_{1vw}^{\max}$ .

```

compute.T1 <- function(X,Y){
  N <- nrow(X)
  J <- ncol(X)
  M <- array(0,c(N,N,J))
  for (j in 1:J){M[, ,j] <- outer(Y[,j],Y[,j],"==")}
  XX <- array(0,c(N,N,J))
  for (j in 1:J){XX[, ,j] <- outer(X[,j],X[,j])}
  T1 <- apply(M * (1 - XX),c(1,2),sum)
  diag(T1) <- 0
  return(T1)
}

```

Figure 1: R function `compute.T1` for computing  $T_1$ .

## 2.1 Computation of $T_1$ .

Equation 1\* shows that  $T_{1_{vw}}$  is computed as

$$T_{1_{vw}} = \sum_{j=1}^J [m_{vw,j} \times (1 - X_{vj} \times X_{wj})]. \quad (2)$$

The R code for computing  $T_1$  using  $\mathbf{Y}$  and  $\mathbf{X}$  is given in Figure 1. Note that by definition  $T_{1_{vv}} = 0$ .

## 2.2 Approximation of $E(T_{1_{vw}})$

For simplicity, we assume that the items are interchangeable and that they have  $m$  options of which 1 is correct and the other  $m - 1$  are incorrect. This renders the solution an approximation of the real expected value but circumvents tedious calculations. The estimated probability that Examinee  $v$  chooses the correct option is  $P_v = X_{v+}/J$ . The estimated probability that Examinee  $v$  chooses a particular incorrect option  $Q_v = \frac{1-P_v}{m-1}$ . First, consider the  $U$  unique items: The estimated probabilities that an examinee pair  $(v, w)$  has a suspicious matching pair-score are

1.  $P_v \times Q_w$  for choosing the option that produces pair-score  $(1, 0)$ ;
2.  $Q_v \times P_w$  for choosing the option that produces pair-score  $(0, 1)$ ; and
3.  $(m - 2) \times Q_v \times Q_w$  for choosing one of the remaining  $m - 2$  options that produces pair-score  $(0, 0)$ .

Therefore, the expected number of suspicious pair-scores on the unique items is

$$E(U) = U \times [P_v Q_w + Q_v P_w + (m - 2) \times Q_v Q_w]$$

Second, consider the  $J - U$  common items: The estimated probabilities that an examinee pair  $(v, w)$  has a suspicious matching pair-score are

1.  $(m - 1) \times Q_v \times Q_w$  for choosing one of the  $m - 1$  incorrect options that produces a pair-score  $(0, 0)$ .

Therefore, the expected number of suspicious pair-scores on the common items is

$$E(J - U) = (J - U)(m - 1)Q_v Q_w.$$

Finally,  $E(T)_{1_{vw}} = E(U) + E(J - U)$ .

The procedure for approximating  $E(T_{1_{vw}})$  for item scores  $\mathbf{X}$ , vector  $\mathbf{v}$  and boolean vector  $\mathbf{u}$  and  $m$  answer categories is given in Figure 2.

```

exp.T1.function <- function(X.v,X.w,J,U,m){
  P.v <- X.v/J
  P.w <- X.w/J
  Q.v <- (1-P.v)/(m-1)
  Q.w <- (1-P.w)/(m-1)
  E.U <- U * ((m-2) * Q.v * Q.w + P.v * Q.w + Q.v * P.w)
  E.C <- (J-U) * (m-1) * Q.v * Q.w
  return(E.C + E.U)
}

approximate.ET1 <- function(X,v=one.version,u=all.common,m=4){
  J <- ncol(X)
  N <- nrow(X)
  one.version <- rep(1,N)
  all.common <- rep(F,J)
  V <- outer(v,v,"==")
  X. <- X %*% matrix(1,nrow=J)
  length.u <- length(u[u==F])

  ET1.same <- matrix(0,J+1,J+1)
  for(i in 0:J) for(j in 0:J) ET1.same[i+1,j+1] <- exp.T1.function(i,j,J,0,m)

  ET1.diff <- matrix(0,J+1,J+1)
  for(i in 0:J) for(j in 0:J) ET1.diff[i+1,j+1] <- exp.T1.function(i,j,J,length.u,m)

  ET1 <- matrix(0,N,N)
  for (i in 1:N) for(j in 1:N){
    ET1[i,j] <- ifelse(V[i,j], ET1.same[X.[i]+1,X.[j]+1],ET1.diff[X.[i]+1,X.[j]+1])
  }
  diag(ET1) <- 0
  return(ET1)
}

```

Figure 2: R function `approximate.ET1` for approximating  $E(T_1)$ . Internal function `exp.T1.function` is used in `approximate.ET1`.

### 2.3 Computation of $T_{1vw}^{\max}$

Assume that a test contains  $U$  unique items and  $J-U$  common items. Also, without loss of generality, assume that  $X_{v+} = L$  and  $X_{w+} = K$  where  $L \geq K$ .  $T_{1vw}$  is an unweighed count of the following suspicious pair-scores in case of a match: (1)  $X_{vw,j} = (1, 0)$  for unique items only; (2)  $X_{vw,j} = (0, 1)$  for unique items only; and (3)  $X_{vw,j} = (0, 0)$  both for unique and common items.

Finding the  $T_{1vw}^{\max}$  can be regarded as finding two response patterns  $\mathbf{x}_v$  (consisting of  $L$  ones and  $J-L$  zeroes) and  $\mathbf{x}_w$  (consisting of  $K$  ones and  $J-K$  zeroes) with as many suspicious pair-scores as possible.

- First, find the maximum number of matching pair-scores  $(1, 0)$  in the responses to the  $U$  unique items. Let this number be denoted  $A$ . Note that  $A$  cannot exceed the number of unique items  $U$ ,  $A$  cannot exceed  $L$  (the number correct by Examinee  $v$ ), and  $A$  cannot exceed  $J-K$  (the number incorrect by Examinee  $w$ ). Hence  $A = \min(U, L, J-K)$ .
- Second, find the maximum number of matching pair-scores  $(0, 1)$  in the responses to the remaining  $U-A$  unique items. Let this number be denoted  $B$ . Note that  $B$  cannot exceed the number of remaining unique items  $U-A$ ,  $B$  cannot exceed  $K$  (the number correct by Examinee  $w$ ), and  $B$  cannot exceed  $J-L$  (the number incorrect by Examinee  $v$ ). Hence,  $B = \min(U-A, J-L, K)$ .
- Third, find the maximum number of matching pair-scores  $(0, 0)$  in the responses to the remaining  $J-A-B$  items. Let this number be denoted  $C$ . Note that  $C$  cannot exceed the number of remaining items  $J-A-B$ ,  $C$  cannot exceed  $J-B-L$  (the number incorrect by Examinee  $v$  minus the number incorrect that were already taken in the second step), and  $C$  cannot exceed  $J-A-K$  (the number incorrect by Examinee  $w$  minus the number incorrect that were already taken in the first step). Hence,  $C = \min(J-A-B, J-B-L, J-A-K)$

The maximum of  $T_{1vw}$  equals  $T_{1vw}^{\max} = A + B + C$ . The function for computing  $T_{1vw}^{\max}$  for item scores  $\mathbf{X}$ , vector  $\mathbf{v}$ , and boolean vector  $\mathbf{u}$  is shown in Figure 3.

### 2.4 Computation of $\tau_{1vw}$

The R functions for computing  $T_1$  (Figure 1), approximating  $E(T_1)$  (Figure 2), and computing  $T_1^{\max}$  (Figure 3) should be combined to compute  $\tau_1$  (Figure 4).

## 3 The computation of $\tau_2$ .

Equation 4\* shows that for examinee pair  $(v, w)$ ,  $\tau_2$  is computed as

$$\tau_{2vw} = \frac{T_{2vw} - E(T_{2vw})}{T_{2vw}^{\max} - E(T_{2vw})}. \quad (3)$$

Equation 3 contains three statistics:  $T_{2vw}$ ,  $E(T_{2vw})$ , and  $T_{2vw}^{\max}$ .

### 3.1 Computation of $T_2$ .

Equation 3\* shows that  $T_{2vw}$  is computed as

$$T_{2vw} = \sum_{j=1}^J m_{vw,j} \times [2X_{wj} \times (1 - X_{vj}) + (1 - X_{vj}) \times (1 - X_{wj})]. \quad (4)$$

The R code for computing  $T_2$  using  $\mathbf{Y}$  and  $\mathbf{X}$  is given in Figure 5. Note that by definition  $T_{2vv} = 0$ .

```

max.T1.function <- function(L,K,J,U){
  if (K > L){G <- L; L <- K; K <- G}
  A <- min(U,L,J-K)
  B <- min(U-A,K,J-L)
  C <- min(J-A-B,J-L-B,J-K-A)
  return(A + B + C)
}

compute.MT1 <- function(X,v=one.version,u=all.common){
  J <- ncol(X)
  N <- nrow(X)
  one.version <- rep(1,N)
  all.common <- rep(F,J)
  V <- outer(v,v,"==")
  X. <- X %%% matrix(1,nrow=J)
  length.u <- length(u[u==F])

  MT1.diff <- matrix(0,J+1,J+1)
  for(i in 0:J) for(j in 0:J) MT1.diff[i+1,j+1] <- max.T1.function(i,j,J,length.u)

  MT1.same <- matrix(0,J+1,J+1)
  for(i in 0:J) for(j in 0:J) MT1.same[i+1,j+1] <- max.T1.function(i,j,J,0)

  MT1 <- matrix(0,N,N)
  for (i in 1:N) for(j in 1:N){
    MT1[i,j] <- ifelse(V[i,j], MT1.same[X.[i]+1,X.[j]+1],MT1.diff[X.[i]+1,X.[j]+1])
  }
  return(MT1)
}

```

Figure 3: R function `compute.MT2` for computing  $T_1^{\max}$ . Internal function `max.T1.function` is used in `compute.MT1`.

```

T1 <- compute.T1(X,Y)
ET1 <- approximate.ET1(X,v,u,m)
MT1 <- compute.MT1(X,v,u)
tau1 <- (T1 - ET1)/(MT1 - ET1)

```

Figure 4: R code for computing  $\tau_1$ .

```

compute.T2 <- function(X,Y){
  N <- nrow(X)
  J <- ncol(X)
  M <- array(0,c(N,N,J))
  for (j in 1:J){M[, ,j] <- outer(Y[,j],Y[,j],"==")}
  XX. <- array(0,c(N,N,J))
  for (j in 1:J){XX.[:, ,j] <- outer(X[,j],1-X[,j])}
  X.X. <- array(0,c(N,N,J))
  for (j in 1:J){X.X.[:, ,j] <- outer(1-X[,j],1-X[,j])}
  T2 <- t(apply(M * (2 * XX. + X.X.),c(1,2),sum))
  diag(T2) <- 0
  return(T2)
}

```

Figure 5: R function `compute.T2` for computing  $T_2$ .

### 3.2 Approximation of $E(T_{2vw})$

The approximation of  $E(T_{2vw})$  follows the same logic as the approximation of  $E(T_{1vw})$  (section 2.2). However, now the counts are weighed so that

$$\begin{aligned}
E(U) &= U [0 \times P_v Q_w + 2Q_v P_w + (m - 2)Q_v Q_w] \\
&= U [2Q_v P_w + (m - 2)Q_v Q_w]
\end{aligned}$$

and

$$E(J - U) = (J - U) \frac{1}{2} (m - 1) Q_v Q_w$$

Finally,  $E(T_{2vw}) = E(U) + E(J - U)$ . The procedure for approximating  $E(T_{2vw})$  for item scores  $\mathbf{X}$ , vector  $\mathbf{v}$  and boolean vector  $\mathbf{u}$  and  $m$  answer categories is given in Figure 6.

### 3.3 Computation of $T_{2vw}^{\max}$

Similarly to computing  $T_{1vw}^{\max}$ , assume that a test contains  $U$  unique items and  $J - U$  common items. Examinee  $v$  is now considered the copier and examinee  $w$  is considered the source. Also, assume that  $X_{v+} = L$  and  $X_{w+} = K$ .  $T_{2vw}$  is a weighed count of the following suspicious matching pair-scores  $X_{vw,j}$ : (1)  $X_{vw,j} = (0, 1)$  for unique items with weight 2 and (2)  $X_{vw,j} = (0, 0)$  both for unique and common items with weight 1. Finding the  $T_{2vw}^{\max}$  can be regarded as finding two response patterns  $\mathbf{x}_v$  (consisting of  $L$  ones and  $J - L$  zeroes) and  $\mathbf{x}_w$  (consisting of  $K$  ones and  $J - K$  zeroes), where the sum of the weighed suspicious pair-scores is as high as possible.

- First, find the maximum number of matching pair-scores  $(0, 1)$  in the responses to the  $U$  unique items. Let this number be denoted  $A$ . Note that  $A$  cannot exceed the number of unique items  $U$ ,  $A$  cannot exceed  $K$  (the number correct by Examinee  $w$ ), and  $A$  cannot exceed  $J - L$  (the number incorrect by Examinee  $v$ ). Hence  $A = \min(U, K, J - L)$ .
- Second, find the maximum number of matching pair-scores  $(0, 0)$  in the responses to the remaining  $J - A$  items. Let this number be denoted  $B$ . Note that  $B$  cannot exceed the number of remaining items  $J - A$ ,  $B$  cannot exceed  $J - K$  (the number incorrect by Examinee  $w$ ), and  $B$  cannot exceed  $J - A - L$  (the number incorrect by Examinee  $v$  minus the number incorrect that were already taken in the first step); the last inequality makes the inequality  $B \leq J - A$  redundant. Hence,  $B = \min(J - K, J - A - L)$ .

```

exp.T2.function <- function(X.v,X.w,J,U,m){
  P.v <- X.v/J
  P.w <- X.w/J
  Q.v <- (1-P.v)/(m-1)
  Q.w <- (1-P.w)/(m-1)
  E.U <- U * ((m-2) * 0.5 * Q.v * Q.w + Q.v * P.w)
  E.C <- (J-U) * 0.5 * (m-1) * Q.v * Q.w
  return(2 * (E.C + E.U))
}

approximate.ET2 <- function(X,v=one.version,u=all.common,m=4){
  J <- ncol(X)
  N <- nrow(X)
  one.version <- rep(1,N)
  all.common <- rep(F,J)
  V <- outer(v,v,"==")
  X. <- X %%% matrix(1,nrow=J)
  length.u <- length(u[u==F])

  ET2.same <- matrix(0,J+1,J+1)
  for(i in 0:J) for(j in 0:J) ET2.same[i+1,j+1] <- exp.T2.function(i,j,J,0,m)

  ET2.diff <- matrix(0,J+1,J+1)
  for(i in 0:J) for(j in 0:J) ET2.diff[i+1,j+1] <- exp.T2.function(i,j,J,J-length.u,m)

  ET2 <- matrix(0,N,N)
  for (i in 1:N) for(j in 1:N){
    ET2[i,j] <- ifelse(V[i,j], ET2.same[X.[i]+1,X.[j]+1],ET2.diff[X.[i]+1,X.[j]+1])
  }
  diag(ET2) <- 0
  return(ET2)
}

```

Figure 6: R function `approximate.ET2` for approximating  $E(T_2)$ . Internal function `exp.T2.function` is used in `approximate.ET2`.

```

max.T2.function <- function(L,K,J,U){
  A <- min(U, K, J-L)
  B <- min(J-K,J-A-L)
  return(2 * A + B)
}

compute.MT2 <- function(X,v=one.version,u=all.common){
  J <- ncol(X)
  N <- nrow(X)
  one.version <- rep(1,N)
  all.common <- rep(F,J)
  V <- outer(v,v,"==")
  X. <- X %%% matrix(1,nrow=J)
  length.u <- length(u[u==F])

  MT2.diff <- matrix(0,J+1,J+1)
  for(i in 0:J) for(j in 0:J) MT2.diff[i+1,j+1] <- max.T2.function(i,j,J,J-length.u)

  MT2.same <- matrix(0,J+1,J+1)
  for(i in 0:J) for(j in 0:J) MT2.same[i+1,j+1] <- max.T2.function(i,j,J,0)

  MT2 <- matrix(0,N,N)
  for (i in 1:N) for(j in 1:N){
    MT2[i,j] <- ifelse(V[i,j], MT2.same[X.[i]+1,X.[j]+1],MT2.diff[X.[i]+1,X.[j]+1])
  }
  return(MT2)
}

```

Figure 7: R function `compute.MT2` for computing  $T_2^{\max}$ . Internal function `max.T2.function` is used in `compute.MT2`.

The maximum of  $T_{2vw}$  equals  $T_{2vw}^{\max} = 2A + B$ . The procedure for computing  $T_{2vw}^{\max}$  for item scores  $\mathbf{X}$ , vector  $\mathbf{v}$ , and boolean vector  $\mathbf{u}$  is shown in Figure 7.

### 3.4 Computation of $\tau_{2vw}$

The R functions for computing  $T_2$  (Figure 5), approximating  $E(T_2)$  (Figure 6), and computing  $T_2^{\max}$  (Figure 7) should be combined to compute  $\tau_2$  (Figure 8).

## 4 The computation of $\tau_1^*$ and $\tau_2^*$ .

According to equations 6\* and 7\*,  $\tau_1^*$  and  $\tau_2^*$  are computed by adding a function of the number of Guttman errors on the matching items and the maximum number of Guttman errors on the matching

```

T2 <- compute.T2(X,Y)
ET2 <- approximate.ET2(X,v,u,m)
MT2 <- compute.MT2(X,v,u)
tau2 <- (T2 - ET2)/(MT2 - ET2)

```

Figure 8: R code for computing  $\tau_2$ .

```

is.one <- function(x,y){ifelse(x==1 & y==1,T,F)}

compute.G <- function(X,Y,v=one.version){
  N <- nrow(X)
  J <- ncol(X)
  one.version <- rep(1,N)
  # TVs = number of different test versions
  TVs <- length(unique(v))
  item.order <- list()
  Nv <- list()
  G <- matrix(0,N,N)
  for (i in 1:TVs){
    Nv[[i]] <- length(v[v==i])
    item.order[[i]] <- rev(order(rep(1/Nv[[i]],Nv[[i]]) %*% X[v==i,] + rnorm(J,0,1e-10)))
    X <- X[,item.order[[i]]]
    Y <- Y[,item.order[[i]]]
    # M = matching response options
    M <- array(0,c(N,N,J))
    for (j in 1:J){M[,,j] <- outer(Y[,j],Y[,j],"==")}
    # A = both item scores equal to 1
    A <- array(0,c(N,N,J))
    for (j in 1:J){A[,,j] <- outer(X[,j],X[,j],is.one)}
    # M = matching item scores equal to 1
    M <- M*A
    # Diagonal is tot number of Guttman errors. Off-diagonal elements are the
    # number of Guttman errors of row examinee minus the number of Guttman errors
    # of row examinee's obtained on the nonmatching items with column examinee.
    for (i in which(v==i)) for (j in 1:N){
      G[i,j] <- sum(X[i,]* M[i,j,] * cumsum(abs(X[i,]-1)))
    }
  }
  diag(G) <- 0
  return(G)
}

```

Figure 9: R function `compute.G` for computing  $G_v$ . Internal function `is.one` is used in `compute.G`.

items to  $\tau_1$  and  $\tau_2$ , respectively; that is,

$$\tau_{1vw}^* = \tau_{1vw} + \frac{1}{2} \left[ \frac{G_v}{G_v^{\max}} \right]; \quad (5)$$

and

$$\tau_{2vw}^* = \tau_{2vw} + \frac{1}{2} \left[ \frac{G_v}{G_v^{\max}} \right]. \quad (6)$$

#### 4.1 Computation of $G_{v|w}$

$G_{v|w}$  is the number of Guttman errors of respondent  $v$  made considering the matching items with respondent  $w$ . The R code is given in Figure 9.

#### 4.2 Computation of $G_v^{\max}$

Consider a test with  $U$  unique items and  $J-U$  common items. Let the items be ordered and numbered by increasing difficulty. The maximum value of  $G_v$  depends on  $X_{v+}$ . For  $X_+ = t$  ( $t = 0, \dots, J$ ) there

exists an item-score vector  $\mathbf{X}^t$  (consisting of  $t$  ones and  $J - t$  zeroes) producing the maximum value of  $G$  given that  $X_+ = t$ . The algorithm for finding  $\mathbf{X}^t$  proceeds as follows. First, for  $t = 0$ , the zero vector  $\mathbf{X}^0 = \mathbf{0}$  yields  $G^{\max} = 0$  for  $X_+ = 0$ . Second, in a stepwise procedure, the 0s in  $\mathbf{X}^0$  are replaced by 1s; one replacement per step. In Step 1, score 0 pertaining the most difficult *common* item is replaced by score 1; the number of Guttman errors from the resulting vector, denoted  $\mathbf{X}^1$ , yields  $G^{\max}$  for  $X_+ = 1$ . In each next step, two cases are compared: Case 1: For the most difficult *common* item which still has score 0, score 0 is replaced by score 1, and the number of Guttman errors is computed. Case 2: For the most difficult *unique* item which still has score 0, score 0 is replaced by score 1, and the number of Guttman errors is computed. In both cases, the number of Guttman errors is computed under the condition that the scores on common items always match with the item of a potential source. If in Step  $t$  the number of Guttman errors in Case 1 is greater than the number of Guttman errors in Case 2, then  $\mathbf{X}^t$  is the item-score vector that resulted from Case 1 and the replacement in Case 2 is cancelled; if the number of Guttman errors in Case 2 is greater than the number of Guttman errors in Case 1, then  $\mathbf{X}^t$  is the item-score vector that resulted from Case 2 and the replacement in Case 1 is cancelled. These steps are repeated until we have a vector  $\mathbf{X}^J$  in which all items have a score 1.

Each step in the algorithm utilizes the property that  $G$  is a scoring function that is *decreasing in transposition* (DT; Rosenbaum, 1987; also, see Emons, 2003) within the set of common items, and DT within the set of unique items. In general, DT means that interchanging a 0 and a 1 score in the item-score vector such that the 1-score is positioned further to the right, has the effect of increasing the scoring function (i.e.,  $G$  in our case). Reversely, interchanging a 0 and a 1 such that the 1 score is positioned further to the left has the effect of decreasing the scoring function. This property is used as follows. In each step of the algorithm, a 1-score is assigned to the most difficult common or unique item, whichever yields the highest number of Guttman errors. Suppose that according to this criterion the most difficult common item was selected. Any other choice for replacing the 0 score of a (less difficult) common item by a 1 would result in a vector that can be obtained from the vector producing the highest  $G$  (i.e., 1-score for the most difficult common item having a score 0) by interchanging 0s and 1s of the common items such that the 1 scores are positioned further to left. Using the DT property, this implies that any other choice for a common item, other than the most difficult one, would produce a smaller value of  $G$ . The same is true when the most difficult unique item led to the highest  $G$ . Then any other choice of a unique item for which the 0 score is replaced by a 1 would yield a smaller value of  $G$ . Thus, once it has been determined whether the most difficult unique item or the most difficult common item produced the highest  $G$ , it also known that this is the highest value that can be obtained because any other choice would lead to a lower value of  $G$ . The R code for computing  $G_v^{\max}$  is given in Figure 10.

### 4.3 Computation of $\tau_1^*$ and $\tau_2^*$

Based on simulations it was found that adding half the ratio of  $G_v$  and  $G_v^{\max}$  to  $\tau_1$  and  $\tau_2$  yielded the best detection rates (Equations 6\* and 7\*). The R code for computing  $\tau_1^*$  and  $\tau_2^*$  is given in Figure 11.

## References

- Emons, W. H. M. (2003). Investigating the local fit of item-score vectors. In H. Yanai, A. Okada, K. Shigemasu, Y. Kano, & J. J. Meulman (Eds.), *New developments in psychometrics* (pp. 289-296). Tokyo: Springer.
- R Development Core Team (2006). *R: A Language and Environment for Statistical Computing* Vienna, Austria: R Foundation for Statistical Computing. Retrieved from <http://www.R-project.org>.

```

max.G.function <- function(X,v=one.version,u=all.common){
  N <- nrow(X)
  J <- ncol(X)
  all.common <- rep(F,J)
  one.version <- rep(1,N)
  TVs <- length(unique(v))
  item.order <- list()
  Nv <- list()
  max.G <- list()
  for (copier in 1:TVs){
    Nv[[copier]] <- length(v[v==copier])
    item.order[[copier]] <- rev(order(rep(1/Nv[[copier]],Nv[[copier]])
      %*% X[v==copier,] + rnorm(J,0,1e-10)))
    max.G[[copier]] <- list()
    for (source in 1:TVs){
      max.G[[copier]][[source]] <- rep(0,J+1)
      if (copier==source){
        u.o <- all.common
      }else{
        u.o <- u[item.order[[copier]]]
      }
      t. <- 1
      common.item <- length(u.o[u.o==F])
      unique.item <- length(u.o[u.o==T])
      X.t <- rep(0,J)
      repeat{
        t. <- t. + 1
        X.t.1 <- X.t.2 <- X.t
        dc <- ifelse(common.item > 0,which(!u.o)[common.item],NA)
        du <- ifelse(unique.item > 0,which(u.o)[unique.item],NA)
        X.t.1[dc] <- 1
        X.t.2[du] <- 1
        G.1 <- ifelse(!is.na(dc),sum((X.t.1 * !u.o) * cumsum(abs(X.t.1-1))),0)
        G.2 <- ifelse(!is.na(du),sum((X.t.2 * !u.o) * cumsum(abs(X.t.2-1))),0)
        if (G.1 >= G.2){
          max.G[[copier]][[source]][t.] <- G.1
          X.t <- X.t.1
          common.item <- common.item - 1
        } else{
          max.G[[copier]][[source]][t.] <- G.2
          X.t <- X.t.2
          unique.item <- unique.item - 1
        }
        if(t.==J+1)break
      }
    }
  }
  return(max.G)
}

compute.MG <- function(X,v=one.version,u=all.common){
  J <- ncol(X)
  N <- nrow(X)
  one.version <- rep(1,N)
  all.common <- rep(F,J)
  max.G <- max.G.function(X,v,u)
  sum <- X %*% rep(1,J)
  MG <- matrix(0,N,N)
  for(i in 1:N) for(j in 1:N){
    MG[i,j] <- max.G[[v[i]]][[v[j]]][sum[i]+1]
  }
  return(MG)
}

```

Figure 10: R function compute.MG for computing  $G_v^{\max}$ . Internal function max.G.function is used in compute.MG.

```
G <- compute.G(X,Y,v)
MG <- compute.MG(X,v,u)
tau1.star <- tau1 + 0.5 * G/MG
tau2.star <- tau2 + 0.5 * G/MG
```

Figure 11: R code for computing  $\tau_1^*$  and  $\tau_2^*$ .

Rosenbaum, P. R. (1987). Probability inequalities for latent scales. *British Journal of Mathematical and Statistical Psychology*, 40, 157-168.

Van der Ark, L. A., Emons, W. H. M., & Sijtsma, K. (2007). *Identifying answer copying using alternate test forms and seat locations in small scale examinations*. (manuscript submitted for publication).