Online Appendix to accompany "New recommendations on the use of R-squared differences in multilevel model comparisons": *r2MLMcomp* R function

r2MLMcomp R function Description:

This function reads in raw data and multilevel model (MLM) parameter estimates from two separate models under comparison (Model A and Model B) and outputs all R^2 measures (reviewed and defined in the manuscript's *print* Appendix A) for both models as well as all ΔR^2 measures (defined in the manuscript's Table 1). Additionally, the function produces side-by-side graphical comparisons of the R^2 measures for Model A vs. B that can be used to visualize changes in each measure across models (see manuscript description of Figure 2 for an example illustration). This function assumes all level-1 predictors are cluster-mean-centered for reasons described in the manuscript. Any number of level-1 and/or level-2 predictors is supported and any of the level-1 predictors can have random slopes. This function can be used with either the hierarchical or the simultaneous model-building approach described in the manuscript (in which Model B is the more complex model for a given pairwise comparison). Our function can also be used with either nested non-nested model comparisons (in which R^2 estimates for Model A are subtracted from those for Model B).

r2MLMcomp R function Input:

data – Dataset with rows denoting observations and columns denoting variables
within_covs_modA – List of numbers corresponding to the columns in the dataset of the level-1 predictors used in the Model A MLM (if none used, set to NULL)
between_covs_modA – List of numbers corresponding to the columns in the dataset of the level-2 predictors used in the Model A MLM (if none used, set to NULL)
random_covs_modA – List of numbers corresponding to the columns in the dataset of the level-1 predictors that have random slopes in the Model A MLM (if no random slopes, set to NULL)
gamma_w_modA – Vector of estimates of the fixed component of slopes for all level-1 predictors for Model A, to be entered in the order of the predictors listed by *within_covs* (if none, set to NULL)
gamma_b_modA – Vector of estimates of the fixed component of the intercept and slopes for all level-2 predictors for Model A, to be entered intercept first followed by level-2 slopes in the order listed by *between_covs*
Tau_modA – random effect covariance matrix for Model A; note that the first row/column denotes the intercept variance and covariances (if intercept is fixed, set all to 0) and each subsequent row/column denotes a given random slope's variance and covariances (to be entered in the order listed by *random_covs*). Variances are on diagonal and covariances are on off-diagonal.
sigma2_modA – level-1 residual variance for Model A
within_covs_modB – List of numbers corresponding to the columns in the dataset of the level-1 predictors used in the Model B MLM (if none used, set to NULL)
between_covs_modB – List of numbers corresponding to the columns in the dataset of the level-2 predictors used in the Model B MLM (if none used, set to NULL)
random_covs_modB – List of numbers corresponding to the columns in the dataset of the level-1 predictors that have random slopes in the Model B MLM (if no random slopes, set to NULL)
gamma_w_modB – Vector of estimates of the fixed component of slopes for all level-1 predictors for Model B, to be entered in the order of the predictors listed by *within_covs* (if none, set to NULL)
gamma_b_modB – Vector of estimates of the fixed component of the intercept and slopes for all

level-2 predictors for Model B, to be entered intercept first followed by level-2 slopes in the order listed by *between_covs*

Tau_modB – random effect covariance matrix for Model B; note that the first row/column denotes the intercept variance and covariances (if intercept is fixed, set all to 0) and each subsequent row/column denotes a given random slope's variance and covariances (to be entered in the order listed by *random_covs*). Variances are on diagonal and covariances are on off-diagonal.

sigma2_modB – level-1 residual variance for Model B

r2MLMcomp **R function Code:**

```
r2MLMcomp <- function(data,within_covs_modA,between_covs_modA,random_covs_modA,
                      gamma_w_modA,gamma_b_modA,Tau_modA,sigma2_modA,
                      within_covs_modB,between_covs_modB,random_covs_modB,
                      gamma_w_modB,gamma_b_modB,Tau_modB,sigma2_modB){
  ##r2MLM function
  r2MLM <- function(data,within_covs,between_covs,random_covs,
                    gamma_w,gamma_b,Tau,sigma2,modelname){
    if(length(gamma_b)>1) gamma <- c(1,gamma_w,gamma_b[2:length(gamma_b)])
   if(length(gamma_b)==1) gamma <- c(1,gamma_w)
   if(is.null(within_covs)==T) gamma_w <- 0
  if(is.null(gamma)) gamma <- 0
  ##compute phi
  phi <- var(cbind(1,data[,c(within_covs)],data[,c(between_covs)]),na.rm=T)
  phi_w <- var(data[,within_covs],na.rm=T)
  if(is.null(within_covs)==T) phi_w <- 0
  phi_b <- var(cbind(1,data[,between_covs]),na.rm=T)
  if(is.null(between_covs)==T) phi_b <- 0
  ##compute psi and kappa
  var_randomcovs <- var(cbind(1,data[,c(random_covs)]),na.rm=T)
    if(length(Tau)>1) psi <- matrix(c(diag(Tau)),ncol=1)
  if(length(Tau)==1) psi <- Tau
    if(length(Tau)>1) kappa <- matrix(c(Tau[lower.tri(Tau)==TRUE]),ncol=1)
  if(length(Tau)==1) kappa <- 0
    v <- matrix(c(diag(var_randomcovs)),ncol=1)
  r <- matrix(c(var_randomcovs[lower.tri(var_randomcovs)==TRUE]),ncol=1)
    if(is.null(random_covs)==TRUE){
   v <- 0
   r <- 0
   m <- matrix(1,ncol=1)
  }
  if(length(random_covs)>0) m <- matrix(c(colMeans(cbind(1,data[,c(random_covs)]),na.rm=T)),ncol=1)
    ##total variance
    totalvar_notdecomp <- t(v)%*%psi + 2*(t(r)%*%kappa) + t(gamma)%*%phi%*%gamma + t(m)%*%Tau%*%m + sigma2
    totalwithinvar <- (t(gamma_w)%*%phi_w%*%gamma_w) + (t(v)%*%psi + 2*(t(r)%*%kappa)) + sigma2
    totalbetweenvar <- (t(gamma_b)%*%phi_b%*%gamma_b) + Tau[1]
    totalvar <- totalwithinvar + totalbetweenvar
    ##total decomp
    decomp_fixed_notdecomp <- (t(gamma)%*%phi%*%gamma) / totalvar
    decomp_fixed_within <- (t(gamma_w)%*%phi_w%*%gamma_w) / totalvar
    decomp_fixed_between <- (t(gamma_b)%*%phi_b%*%gamma_b) / totalvar
    decomp_fixed <- decomp_fixed_within + decomp_fixed_between
    decomp_varslopes <- (t(v)%*%psi + 2*(t(r)%*%kappa)) / totalvar
    decomp_varmeans <- (t(m)%*%Tau%*%m) / totalvar
    decomp_sigma <- sigma2/totalvar
    ##within decomp
    decomp_fixed_within_w <- (t(gamma_w)%*%phi_w%*%gamma_w) / totalwithinvar
    decomp_varslopes_w <- (t(v)%*%psi + 2*(t(r)%*%kappa)) / totalwithinvar
    decomp_sigma_w <- sigma2/totalwithinvar
    ##between decomp
    decomp_fixed_between_b <- (t(gamma_b)%*%phi_b%*%gamma_b) / totalbetweenvar
    decomp_varmeans_b <- Tau[1] / totalbetweenvar
    ##measures
     R2_f <- decomp_fixed
   R2_f1 <- decomp_fixed_within
   R2_f2 <- decomp_fixed_between
```

```
    R2_fv <- decomp_fixed + decomp_varslopes
    R2_fvm <- decomp_fixed + decomp_varslopes + decomp_varmeans
    R2_v <- decomp_varslopes
    R2_m <- decomp_varmeans
    R2_f_w <- decomp_fixed_within_w
    R2_f_b <- decomp_fixed_between_b
    R2_fv_w <- decomp_fixed_within_w + decomp_varslopes_w
    R2_v_w <- decomp_varslopes_w
    R2_m_b <- decomp_varmeans_b
     decomp_table <- matrix(c(decomp_fixed_within,decomp_fixed_between,decomp_varslopes,decomp_varmeans,decomp_sigma,
                        decomp_fixed_within_w,"NA",decomp_varslopes_w,"NA",decomp_sigma_w,
                        "NA",decomp_fixed_between_b,"NA",decomp_varmeans_b,"NA"),ncol=3)
    rownames(decomp_table) <- c("fixed, within","fixed, between","slope variation","mean variation","sigma2")
    colnames(decomp_table) <- c("total","within","between")
    R2_table <- matrix(c(R2_f1,R2_f2,R2_v,R2_m,R2_f,R2_fv,R2_fvm,
                      R2_f_w,"NA",R2_v_w,"NA","NA",R2_fv_w,"NA",
                      "NA",R2_f_b,"NA",R2_m_b,"NA","NA","NA")
                     ,ncol=3)
    rownames(R2_table) <- c("f1","f2","v","m","f","fv","fvm")
    colnames(R2_table) <- c("total","within","between")
  ##barchart
   contributions_stacked <- matrix(c(decomp_fixed_within,decomp_fixed_between,decomp_varslopes,decomp_varmeans,decomp_sigma,
                                 decomp_fixed_within_w,0,decomp_varslopes_w,0,decomp_sigma_w,
                                 0,decomp_fixed_between_b,0,decomp_varmeans_b,0),5,3)
    colnames(contributions_stacked) <- c("total","within","between")
    rownames(contributions_stacked) <- c("fixed slopes (within)",
                                   "fixed slopes (between)",
                                   "slope variation (within)",
                                   "intercept variation (between)",
                                   "residual (within)")
    barplot(contributions_stacked, main=paste0("Decomposition of Scaled Variance, Model ",modelname), horiz=FALSE,
          ylim=c(0,1),col=c("darkred","steelblue","darkred","midnightblue","white"),ylab="proportion of variance",
          density=c(NA,NA,30,40,NA),angle=c(0,45,0,135,0),xlim=c(0,1),width=c(.3,.3))
    legend(.33,-.1,legend=rownames(contributions_stacked),fill=c("darkred","steelblue","darkred","midnightblue","white"),
         cex=.7, pt.cex = 1,xpd=TRUE,density=c(NA,NA,30,40,NA),angle=c(0,45,0,135,0))
   Output <- list(noquote(decomp_table),noquote(R2_table))
  names(Output) <- c("Decompositions","R2s")
  return(Output)
 }
 ##compute decomp for Model A and B
 results_modA <- r2MLM(data,within_covs_modA,between_covs_modA,random_covs_modA,
                  gamma_w_modA,gamma_b_modA,Tau_modA,sigma2_modA,"A")
 decomp_modA <- results_modA$Decompositions
 results_modB <- r2MLM(data,within_covs_modB,between_covs_modB,random_covs_modB,
                  gamma_w_modB,gamma_b_modB,Tau_modB,sigma2_modB,"B")
 decomp_modB <- results_modB$Decompositions
##comparison measures
 delta_f1_t <- as.numeric(decomp_modA[1,1]) - as.numeric(decomp_modB[1,1])
 delta_f2_t <- as.numeric(decomp_modA[2,1]) - as.numeric(decomp_modB[2,1])
 delta_v_t <- as.numeric(decomp_modA[3,1]) - as.numeric(decomp_modB[3,1])
 delta_m_t <-as.numeric(decomp_modA[4,1]) - as.numeric(decomp_modB[4,1])
delta_f1_w <- as.numeric(decomp_modA[1,2]) - as.numeric(decomp_modB[1,2])
 delta_v_w <- as.numeric(decomp_modA[3,2]) - as.numeric(decomp_modB[3,2])
delta_f2_b <- as.numeric(decomp_modA[2,3]) - as.numeric(decomp_modB[2,3])
 delta_m_b <- as.numeric(decomp_modA[4,3]) - as.numeric(decomp_modB[4,3])
delta_f_t <- delta_f1_t + delta_f2_t
 delta_fv_t <- delta_f1_t + delta_f2_t + delta_v_t
 delta_fvm_t <- delta_f1_t + delta_f2_t + delta_v_t + delta_m_t
 delta_f1v_w <- delta_f1_w + delta_v_w
 ##comparison bar charts
contributions_stacked_total <-
matrix(c(as.numeric(decomp_modA[1,1]),as.numeric(decomp_modA[2,1]),as.numeric(decomp_modA[3,1]),as.numeric(decomp_modA[4,1]),as.
numeric(decomp_modA[5,1]),
as.numeric(decomp_modB[1,1]),as.numeric(decomp_modB[2,1]),as.numeric(decomp_modB[3,1]),as.numeric(decomp_modB[4,1]),as.numeric(d
ecomp_modB[5,1])),5,2)
 colnames(contributions_stacked_total) <- c("Model A","Model B")
  barplot(contributions_stacked_total, main="Decomposition of Scaled Total Variance", horiz=FALSE,
       ylim=c(0,1),col=c("darkred","steelblue","darkred","midnightblue","white"),ylab="proportion of variance",
       density=c(NA,NA,30,40,NA),angle=c(0,45,0,135,0),width=c(.3,.3))
  legend(0.26,-.1,legend=c("fixed slopes (within)",
```

```
                        "fixed slopes (between)",
                        "slope variation (within)",
                        "intercept variation (between)",
                        "residual (within)"),fill=c("darkred","steelblue","darkred","midnightblue","white"),
       cex=.7, pt.cex = 1,xpd=TRUE,density=c(NA,NA,30,40,NA),angle=c(0,45,0,135,0))
   contributions_stacked_within <-
matrix(c(as.numeric(decomp_modA[1,2]),0,as.numeric(decomp_modA[3,2]),0,as.numeric(decomp_modA[5,2]),
                                  as.numeric(decomp_modB[1,2]),0,as.numeric(decomp_modB[3,2]),0,as.numeric(decomp_modB[5,2])),5,2)
  colnames(contributions_stacked_within) <- c("Model A","Model B")
  barplot(contributions_stacked_within, main="Decomposition of Scaled Within-Cluster Variance", horiz=FALSE,
        ylim=c(0,1),col=c("darkred","steelblue","darkred","midnightblue","white"),ylab="proportion of variance",
        density=c(NA,NA,30,40,NA),angle=c(0,45,0,135,0),width=c(.3,.3))
  legend(0.28,-.1,legend=c("fixed slopes (within)",
                        "slope variation (within)",
                        "residual (within)"),fill=c("darkred","darkred","white"),
       cex=.7, pt.cex = 1,xpd=TRUE,density=c(NA,30,NA),angle=c(0,0,0))
   contributions_stacked_between <-  matrix(c(0,as.numeric(decomp_modA[2,3]),0,as.numeric(decomp_modA[4,3]),0,
                                   0,as.numeric(decomp_modB[2,3]),0,as.numeric(decomp_modB[4,3]),0),5,2)
  colnames(contributions_stacked_between) <- c("Model A","Model B")
  barplot(contributions_stacked_between, main="Decomposition of Scaled Between-Cluster Variance", horiz=FALSE,
        ylim=c(0,1),col=c("darkred","steelblue","darkred","midnightblue","white"),ylab="proportion of variance",
        density=c(NA,NA,30,40,NA),angle=c(0,45,0,135,0),width=c(.3,.3))
   legend(0.26,-.1,legend=c("fixed slopes (between)",
                        "intercept variation (between)"),fill=c("steelblue","midnightblue"),
       cex=.7, pt.cex = 1,xpd=TRUE,density=c(NA,40),angle=c(45,135))
   ##table of R2 deltas
   R2_modA <- results_modA$R2s
  R2_modB <- results_modB$R2s
   R2_delta <- suppressWarnings(as.numeric(R2_modB) - as.numeric(R2_modA))
  R2_delta <- matrix(R2_delta,7,3)
  colnames(R2_delta) <- colnames(R2_modA)
  rownames(R2_delta) <- rownames(R2_modA)
   Output <- list(R2_modA,R2_modB,R2_delta)
  names(Output) <- c("Model A R2s","Model B R2s","R2 differences, Model B - Model A")
    return(Output)
  }
```

r2MLMcomp R function Example Input:

```
#NOTE: estimates in the input represent hypothetical results for a comparison between a random slope model with two
#level-1 predictors and one level-2 predictor (Model A) and a model that adds an additional level-2 predictor (Model B);
#in practice a user would have previously obtained these input estimates by fitting their models in MLM software;
#additionally, the input consists of hypothetical predictor data, whereas in practice a user would read-in their actual data

data <- matrix(NA,100,4)
xs <- mvrnorm(n=100,mu=c(0,0),Sigma=matrix(c(2,.75,.75,1.5),2,2))
ws <- mvrnorm(n=10,mu=c(0,2),Sigma=matrix(c(1,.5,.5,2),2,2))
data[,1:2] <- xs
for (i in seq(10)){
  data[(10*(i-1)+1):(i*10),3] <- ws[i,1]
  data[(10*(i-1)+1):(i*10),4] <- ws[i,2]
  data[(10*(i-1)+1):(i*10),1] <- data[(10*(i-1)+1):(i*10),1] - mean(data[(10*(i-1)+1):(i*10),1])
  data[(10*(i-1)+1):(i*10),2] <- data[(10*(i-1)+1):(i*10),2] - mean(data[(10*(i-1)+1):(i*10),2])
}
r2MLMcomp(data,within_covs_modA=c(1,2),between_covs_modA=c(3),random_covs_modA=c(1,2),
        gamma_w_modA=c(2.5,-1),gamma_b_modA=c(1,4),Tau_modA=matrix(c(8,1,.75,1,1,.25,.75,.25,.5),3,3),sigma2_modA=10,
        within_covs_modB=c(1,2),between_covs_modB=c(3,4),random_covs_modB=c(1,2),
        gamma_w_modB=c(2.5,-1),gamma_b_modB=c(1,4,1),Tau_modB=matrix(c(6.5,.5,.25,.5,1,.25,.75,.25,.5),3,3),sigma2_modB=10)
```